VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky

DIPLOMOVÁ PRÁCE

2014 **Tomáš Trval**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Autonomní kontrola stavu přejezdových závor pomocí počítačového vidění

Autonomous control of level crossing barrier using computer vision

2014 **Tomáš Trval**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Tomáš Trval**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Autonomní kontrola stavu přejezdových závor pomocí počítačového vidění
Autonomous Control of Level Crossing Barrier using Computer Vision

Zásady pro vypracování:

Bezpečnost železničních přejezdů je již mnoho let velmi diskutované téma. Součásti bezpečnostního systému bývá mnoho modulů, které sledují celkový stav přejezdových zařízení a zajišťují tak jeho bezpečný provoz. Tato práce se zabývá sledováním stavu přejezdových závor.
Cílem této práce je vytvoření systému schopného měřit pozici a stav této bariéry.

1. Nastudovat metody detekce objektu v obraze.
2. Vytvořit detektor závory železničního přejezdu.
3. Ověřit funkci na testovacích datech.

Seznam doporučené odborné literatury:

Podle pokynů vedoucího diplomové práce.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Karel Mozdřeň**

Datum zadání: 01.09.2013
Datum odevzdání: 07.05.2014

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Podpis: .....

Poděkování

Chtěl bych poděkovat svému vedoucímu diplomové práce Ing. Karlu Mozdřeňovi za odborné vedení, pomoc a rady při zpracování této práce. Mé poděkování patří též akciové společnosti Trakce a.s. za poskytnutá data.

Abstrakt

Tato diplomová práce se zabývá problematikou autonomní kontroly stavu přejezdových závor pomocí počítačového vidění. V úvodu je vysvětlena motivace k vytvoření této práce a představen problém, kterým se bude zabývat. Teoretická část se věnuje popisu metod použitých při návrhu detektoru. Dále je shrnuto čemu a proč se věnuje obor detekce objektů v obrazech a jsou uvedeny příklady aplikací z tohoto oboru. K detekci přejezdových závor byly navrhnuty a implementovány dva přístupy. Hlavní část práce popisuje jejich princip a vývoj. V kapitole experimentů je provedeno měření úspěšnosti vytvořených detektorů. Závěrem práce jsou shrnuty získané zkušenosti a porovnány vytvořené detektory.

Klíčová slova: *Počítačové vidění, Zpracování obrazu, Detekce závor, Železníční přejezd*

Abstract

This diploma thesis deals with the problem of autonomous control of level crossing barrier using computer vision. The introduction explains the motivation to create this work and it presents the problem to be dealt with. The theoretical part describes the methods used in the design of the detector. Next section summarizes what and why the field of object recognition in images is used for and examples are given of applications in this field. Two approaches have been designed and implemented to detect the level crossing barriers. The main part of the thesis describes their principle and development. Measuring the success of a created detectors is done in chapter of experiments. The conclusion summarizes the experience gained. Created detectors are compared in this chapter too.

Keywords: *Computer vision, Image processing, Barrier detetion, Level crossing*

Seznam použitých zkratek a symbolů

LIDAR	- light detection and ranging
GPU	- graphics processing unit
IR	- infrared
MCC	- Matthews correlation coefficient

Obsah

1 Úvod

V dnešní době je oblast počítačového vidění rychle se rozvíjející obor. Za tímto prudkým vzestupem stojí levná výroba kamer i dostatečně výkonných čipů. Počítače tak bývají vybaveny lepším zrakem než mají lidé a také jsou schopny zpracovat větší množství informací. Se speciálními kamerami mohou vidět i spektra obrazu, které není lidský zrak schopen zachytit. Například infračervené či ultrafialové záření. Problémem u počítačů tedy není, že by špatně viděly. Nedokáží jen zaznamenaným datům porozumět. Mají tedy zrak, ale nevidí. Vidění je proces při němž nejen snímáme obrazovou informaci z okolí, ale zároveň z ní extrahujeme důležité informace o prostředí, předmětech a osobách ve scéně. Tyto informace jsou dále využívány pro správné vyhodnocení situace pozorovatele. Podobných výsledků se lidé nyní snaží docílit i u počítačů pomocí vhodných algoritmů.

S využitím dostupným technologií jsou dnes ve velkoměstech vytvářeny centrální integrované systémy bezpečnostních i krizových složek. Jejich součástí bývá i kamerový systém, rozmístěný na klíčových bodech jako jsou veřejné budovy, křižovatky a dopravní tepny. Tento trend také proniká do dalších systémů spojených především z dopravou a bezpečností. Příkladem mohou být asistenční systémy v chytrých automobilech (obr. 1). Příklad ukazuje práci detektoru, který v noci zjišťuje přítomnost zvířat a osob na vozovce. Včasným informováním řidiče tak lze předejít vážným zraněním a ztrátám na životech.

Obrázek 1: Využití počítačového vidění pro zvýšení bezpečnosti dopravy, zdroj [10]

V posledních letech je velmi diskutované téma bezpečnost železničních přejezdů. Některé z nehod na přejezdech bývají způsobeny selháním spouštěcího mechanismu závor. Pro zachycení těchto selhání se nabízí možnost vizuálního potvrzení stavu závory. V této diplomové práci se zabývám tvorbou detektoru přejezdových závor pro autonomní kontrolu jejich stavu. První kapitoly jsou věnovány obrazu, chybám v obraze a způsobům jejich minimalizace. Též popisuji vybrané metody detekce objektů. Z nich jsem využil thresholding a ruční označení předpokládané pozice objektu.

Další sekce popisuje, co je to detekce objektů, proč se detekce objektů používá a k čemu slouží. Jako vzorové kategorie účelů detekce jsem vybral aplikace pro rozpoznávání objektů a analýzu pohybu entit ve video sekvencích. Ke zmíněným kategoriím aplikací jsem také uvedl

konkrétní příklady z praxe. Jedním příkladem je dobře známá detekce obličejů na fotografiích. Pro přiblížení tématu dopravy jsem zvolil jako druhý příklad detekci okrajových čar vozovky.

Hlavní částí této práce je tvorba detektoru přejezdových závor pro autonomní kontrolu jejich stavu. Za tímto účelem byly vytvořeny a porovnány dvě metody. Jedna prohledává celý vstupní obraz. Druhá vyžaduje označení pozice na níž se má spuštěná závora nacházet a ke klasifikaci výskytu závory na vybraném místě využívá strojového učení. Nejprve jsou obě metody rozebrány a popsány. Závěrem kapitoly jsou pak přístupy obou metod porovnány. Zkoumáním spolehlivosti obou metod se pak podrobněji zabývá kapitola experimentů.

Sekce experimentů je věnována porovnání úspěšnosti vytvořených metod. Nejprve jsou popsána vstupní data. Těmi je sekvence snímků zaznamenaná kamerou na železničním přejezdu. Tato data jsou pro oba detektory stejná. Vstupní data nejprve klasifikuji ručně a následně i pomocí jednotlivých detektorů. Získané výsledky jsou analyzovány a následně je provedena diskuse úspěšných i problémových případů. V sekci závěru jsou shrnuty dosažené výsledky, možnosti obou detektorů a nově získané zkušenosti.

2 Teoretický základ

V této kapitole je uvedena vybraná teorie týkající se zpracování digitálního obrazu, která je použita při tvorbě detektoru přejezdových závor. První část se věnuje některým vlastnostem vstupního obrazu. Zmiňuji zde chyby, které může vstupní obraz obsahovat a způsoby jejich minimalizace. V druhé části jsou popsány vybrané procesy prováděné při detekování objektů v obraze.

2.1 Vstupní obraz

V dnešní digitální době kdy většina lidí disponuje nějakou formou technologie schopnou zaznamenávat digitální obraz se zdá, že je každému jasné co to vlastně obraz je. Pro běžného uživatele se jedná o množinu pixelů, které dohromady tvoří obrazový záznam nějaké události. Na obraz se dá však dívat mnohem zajímavěji a to jako na dvourozměrnou funkci. V případě digitálního obrazu nespojitou. Obecně ale můžeme uvažovat, že jde o funkci spojitou. tento pohled nám otevírá mnohé možnosti a přístupy, které by nás zřejmě nenapadlo na pouhou množinu bodů aplikovat. Třeba metody z praxe pro práci s elektrickými signály. Například je možné se pokusit o filtrování nízkých či vysokých kmitočtů, práci s fází, amplitudou a podobně [1].

Signál se samozřejmě nemusí omezovat pouze na prostor roviny. Obecně lze uvažovat signály m-rozměrné. Výraz (1) popisuje m-rozměrný signálový prostor φ. Přičemž $f : \Omega \to \gamma$ je matematickou reprezentací signálu. Kde Ω představuje podmnožinu m-rozměrného euklidovského prostoru E^m. Množina γ zase obor hodnot a společně tvoří funkci zobrazení $\Omega \to \gamma$. Přičemž obor hodnot γ může být množina celých čísel pro obrazy ve stupních šedé nebo u barevných obrazů se může jednat o tří-složkový vektor reprezentující hodnoty kanálů R, G, B. Obor hodnot je závislý od operací, které nad obrazem provádíme, či výsledků které dostáváme.

$$\varphi = \{f | f : \Omega \to \gamma\} \tag{1}$$

Stejně jako v každém jiném signálu i v signálu obrazovém se mohou vyskytovat chyby zapříčiněné nekvalitním záznamovým zařízením či rušením v přirozeném prostředí. Ošetření a redukci těchto chyb se budu věnovat v další podkapitole.

2.1.1 Chyby a jejich korekce

V této kapitole se budu věnovat popisu vybraných chyb, které vznikají v obrazech. Jak už jsem uvedl výše, je obraz signál. Tudíž stejně jako například radiový signál může být i on ovlivněn rušením prostředí. Radiový signál je citlivý na některé atmosferické vlivy. Úder blesku poblíž antény se projeví jako praskání v demodulovaném zvukovém signálu. Obecně můžeme tento jev nazvat šum. Dalším problémem mohou být špatné světelné podmínky jako šero, mlha, déšť či sníh. Ty lze přirovnat k velké vzdálenosti mezi radiovým přijímačem a vysílačem. Tehdy je signál už tak slabý, že je obtížné jej v plné kvalitě rekonstruovat nebo dokonce zachytit. Toto má za následek snížení rozsahu intenzity signálu. Tedy v případě obrazu není využitá celá škála barev a obraz je nevýrazný.

2.1.2 Odstranění šumu

Šum, jak už intuitivně chápeme z uvedené analogie s atmosferickými poruchami, je nepravidelný jev. Stejně tak je i jeho rozmístění v obraze náhodné a právě této vlastnosti mohu využít k minimalizaci jeho vlivu. Šum představuje ve svém nejbližším okolí rušivou hodnotu. Liší se tedy od průměrné hodnoty vzorků z nejbližšího okolí. Rušivou odchylku lze snížit pokud místo samotné hodnoty v bodě rušení použiji průměrnou hodnotu barvy blízkého okolí. Navíc mohu jednotlivým vzorkům z okolí přiřadit váhu, která je nepřímo závislá na vzdálenosti vzorku od zpracovávaného bodu. K tomuto účelu lze využít konvoluce, přesněji gaussovského rozmazání. To provede vyhlazení obrazu, viz. [2, str. 137, 150]. Nevýhodou této operace je ztráta ostrosti hran objektů v obraze.

(a) Vstupní obraz (b) Výsledek guassian blur

Obrázek 2: Příklad omezení šumu

Konvoluce je operace zpracovávající dvě funkce (2). Výsledná hodnota v bodě (x, y) je vypočtena jako "vážený průměr"funkce f. Váhovou funkcí, která bývá též nazývána konvoluční jádro, je funkce h. Kvůli záporným znaménkům se při výpočtu funkce h otočí o 180° a posune o vektor (x, y), [1, str. 11]. Při provádění gaussovského rozostření je jádrem konvoluce gaussovská funkce (3). Ze záporného exponentu u souřadnic obou čtverců je patrné, že hodnota váhy klesá se vzdáleností od středu souřadnic této funkce symetricky [2, str. 137, 150].

$$(f * h)(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(a, b)h(x - a, y - b)\,\mathrm{d}a\mathrm{d}b \tag{2}$$

$$G(r, s) = \frac{1}{2\pi\sigma^2} e^{-\frac{r^2+s^2}{2\sigma^2}} \tag{3}$$

Protože pracuji s digitálním obrazem je vhodné uvést předpis diskrétní konvoluce (4). Zde jsou již funkce f a h diskrétními obrazovými signály z prostoru

$Omega = (m,n)|m = 0,1,...,M-1; n =,...,N-1.$

$$(f * h)(m,n) = \frac{1}{MN} \sum_{r=0}^{M-1} \sum_{s=0}^{N-1} f(r,s) \cdot h(m-r, n-s) \tag{4}$$

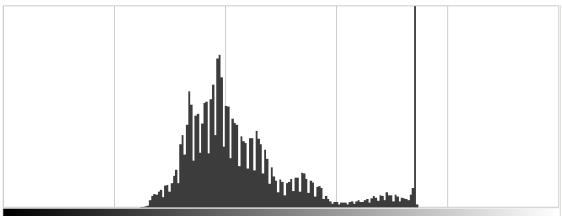
Příklad filtrace šumu (obr 2) se skládá ze dvou obrazů. Vstupní snímek (obr. 2a) obsahuje šum, který by způsoboval problémy při dalším zpracování obrazu. Aby byl tento šum odstraněn je na vstupní obraz aplikováno výše zmíněný gaussovské rozostření. Ve výsledném obraze (obr. 2b) je šum zjevně potlačen a celý obraz je neostrý.

2.1.3 Normalizace obrazu

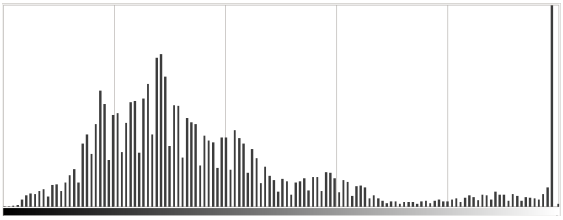
Normalizace obrazu je proces, který mění rozsah intenzity barev v pixelech. Někdy bývá normalizace nazývána roztahování histogramu. Problém malého rosahu intenzity barev nastává například pokud jsou z několika obrazů odebírány vzorky (menší oblasti), které jsou následně navzájem porovnávány. V každém obraze nemusí být stejné světelné podmínky. Každý může být také pořízen jiným přístrojem nebo může být při ukládání obrazu použita ztrátová komprese. Toto ovlivňuje rozsah intenzity barev pixelů odebraných vzorků. Aby byly vzorky vzájemně porovnatelné je třeba sjednotit (normalizovat) tyto rozsahy intenzit [6, str. 85]..

(a) Omezený rozsah intenzity pixelů

(b) Normalizovaný obraz



(c) Histogram omezeného rozsahu intenzity



(d) Histogram normalizovaného obrazu

Obrázek 3: Normalizování obrazu

$$I_{new} = (I - Min)\frac{Max_{new} - Min_{new}}{Max - Min} + Min_{new} \tag{5}$$

Ve výrazu (5) je I_{new} nová hodnota intenzity pro obraz ve stupních šedi. Intenzita původního obrazu je označena I. Dynamický rozsah původního obrazu náleží intervalu $< Min; Max >$. Dynamický rozsah normalizovaného obrazu se nachází v intervalu $< Min_{new}; Max_{new} >$.

V příkladu (obr. 3) je vlevo obraz ve stupních šedi (obr. 3a). Pod ním se nachází jeho histogram (obr. 3c). Z něj je patrno, že obraz nevyužívá celého možného rozsahu intenzity pixelů. Nevyskytují se v něm jasně bílé ani sytě černé barvy. Vpravo (obr. 3b) je ten samý obraz, na kterém byla uplatněna normalizace. Pod ním se nalézá jeho histogram (obr. 3d). Hodnoty intenzity nyní využívají celý možný rozsah intenzity daného formátu.

2.2 Metody detekce objektů

Jednou z hlavních úloh počítačového vidění je rozpoznávání objektů v obraze. Jejím účelem je v obraze objekty nalézt. Jedním ze spolehlivých způsobů nalezení objektů je segmentace. Jádrem tohoto procesu je předpoklad, že hledané objekty mají určitou úroveň jasu či barvu, kterou se výrazně liší od okolí. Vytváří v obraze tedy rozpoznatelnou oblast. Tyto metody lze principiálně rozdělit na prahování, spojování oblastí a dělení oblastí. Pro úplnost uvádím i další metodu a to ruční výběr objektů.

2.2.1 Ruční označení

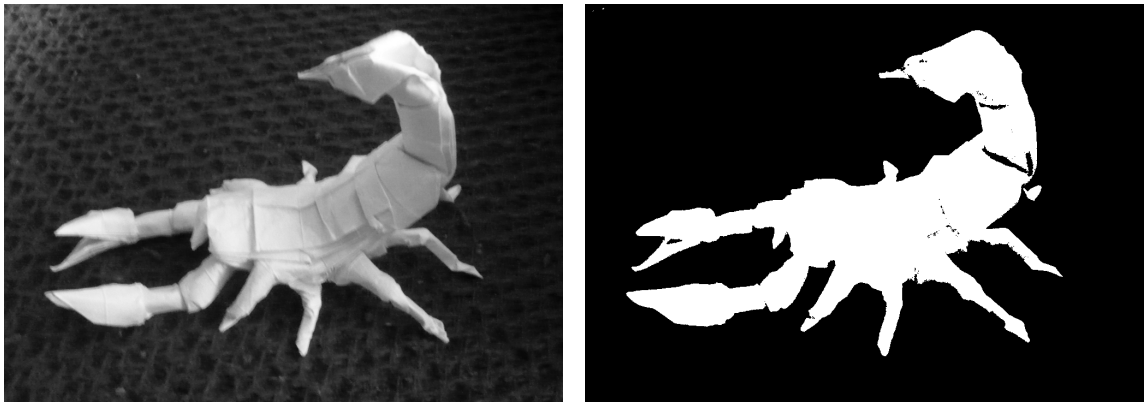
Jak již název napovídá tato metoda vyžaduje jako součást procesu detekování lidský prvek. Na první pohled se tato metoda nemusí zdát vhodnou pro automatizované zpracování obrazu. Při zvážení speciálního případu, kdy je kamera statická a detekovaný objekt se pohybuje po stále stejné dráze může tato metoda představovat elegantní řešení. Je třeba označit hledaný objekt pouze jednou a detektor naučit rozpoznávat stav přítomnosti objektu na dané pozici. Další detekce objektu budou prováděny automatizovaně.

2.2.2 Thresholding

Thresholding neboli prahování je metoda, která detekuje celé oblasti. Je to zřejmě nejjednodušší, ale účinná a rychlá metoda pro nalezení objektů v obraze. Výsledkem prahování je binární obraz, ve kterém nejčastěji hodnota 1 označuje pixel náležící objektu a hodnota 0 náleží pixelu pozadí. Prahování předpokládá v celé oblasti představující jeden objekt konstantní nebo téměř neměnnou hodnotu sledovaného parametru. Parametrem může být jas, sytost, barevný odstín nebo můžeme sledovat i několik parametrů současně. Při prahování označím interval $< p_{min}, p_{max} >$, ve kterém mají hodnoty sledovaného parametru ležet a body splňující tuto podmínku jsou nalezenými body objektu. Taktéž může být vybrána pouze jediná hodnota prahu t. Pak body které mají hodnotu prahu vyšší než t jsou nalezenými body objektu [1, str. 87].

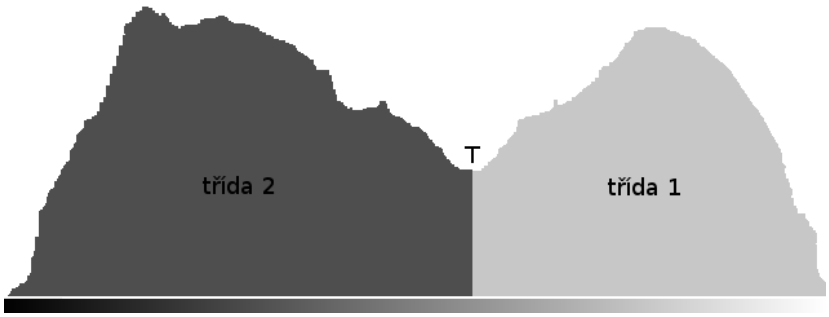
Dále je uveden příklad thresholdingu (obr. 4). Vstupní obraz obsahuje světlý bílý předmět na tmavém pozadí a celý je v odstínech šedé (obr. 4a). Stačí tedy zvolit vhodnou hodnotu prahu a spustit proces prahování jehož výsledkem je binární obraz (obr. 4b).

Samotná volba prahu nebo intervalu je klíčová pro úspěšnou detekci objektů. Často tato hodnota vyplývá ze zadání. Ale stejně tak dobře se může stát, že ji musí volit sám autor detektoru. Při volbě prahu je vhodné prozkoumat tvar histogramu. Pokud má například tvar sedla (obr. 5) je vhodné zvolit hodnotu prahu z oblasti sedlové níže T. Rozdělením vzniknou dvě třídy pixelů, z nichž jedna reprezentuje objekt a druhá pozadí.



(a) Vstupní obraz (b) Výsledek prahování

Obrázek 4: Příklad thresholdingu



Obrázek 5: Sedlová níže

Pro hledání objektů dle odstínu barvy se nabízí i jiné řešení než sedlový bod. Pokud předem známe jaký odstín má objekt mít, lze využít převodu do jiného barevného modelu. V základu uvažujeme, že barevná složka se skládá ze tří kanálů RGB. Pro budoucí detekci objektů v je však vhodnější převést obraz do barevného modelu HSV. Tento model je pro lidské vnímání mnohem přirozenější. Výhodou je, že barevný odstín je volen jen jedním parametrem H a zbylé parametry S, V nastavují sytost a jas. Tedy na změny světelných podmínek jsou citlivé především kanály S a V.

2.2.3 Metoda spojování oblastí

Tato iterativní metoda postupuje od atomických oblastí (pixelů), ze kterých sestavuje spojováním větší a větší oblast. Kritériem zastavení je zde situace, kdy už nelze žádné oblasti spojit. Při procesu spojování hrají důležitou roli kritéria, za kterých se mohou dvě oblasti spojit. Například musí mít podobný odstín či jas. Neměly by být odděleny ostrou hranou, taktéž délka hranice sousedících oblastí může hrát roli. Není zřejmě vhodné spojovat oblasti dotýkající se pouze rohovými body a naopak je žádoucí spojit oblasti sousedící spolu dlouhou hranicí [1, str. 89].

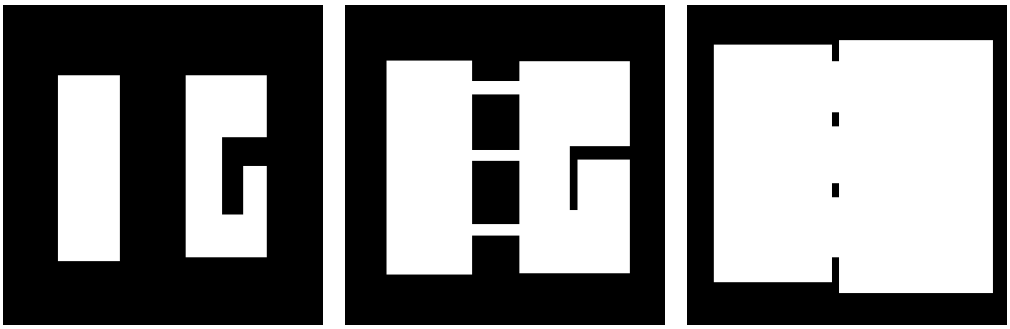
2.2.4 Operace nad binárním obrazem

Zajímavou a přesto ve svém základu jednoduchou a účinnou metodou pro práci s binárními obrazy je matematická morfologie. Vstupem V je binární obraz s detekovanými oblastmi a malý binární obraz představující masku M. Pro tyto vstupy definuje matematická morfologie dvě základní operace: *dilataci* a *erozi*. Princip spočívá v postupném přikládání masky M na pixely vstupního obrazu V.

$$E = V \otimes M\{x, y | M_{x,y} \subseteq V\} \tag{6}$$

$$D = V \oplus M\{x, y | M_{x,y} \cap V \neq \emptyset\} \tag{7}$$

Výraz (6) popisuje vznik erodovaného obrazu E použitím masky M na obraz V. Přičemž pixel náleží objektu pouze tehdy a jen tehdy, když pixely vstupního obrazu překryté jednotkovými pixely masky náleží do objektu. Výraz (7) popisuje vznik dilatovaného obrazu E použitím masky M na obraz V. Přičemž pixel náleží do objektu pouze tehdy a jen tehdy, když alespoň jeden z pixelů vstupního obrazu překrytých jednotkovými pixely masky náleží do objektu. Informace o možnostech matematické morfologie jsem čerpal z textů [1, str. 91, 92] a [7] .



(a) Eroze (b) Originální obraz (c) Dilatace

Obrázek 6: Příklady operací

V příkladu (obr. 6) jsou znázorněny výsledky operací eroze a dilatace na původní binární obraz (obr. 6b). Po provedení eroze (obr. 6a) jsou velké plochy spojené tenkými elementy odděleny, otvory v souvislých plochách zvětšeny a plocha objektů zmenšena. Při provedení dilatace (obr. 6c) jsou objekty, které se nachází blízko sebe spojeny, otvory či zářezy v souvislých plochách zaceleny a plocha objektů zvětšena.

Nepříjemnou vlastností dilatace a eroze je změna velikosti objektů. Tímto neduhem však netrpí další často používané operace otevření a uzavření. Uzavření (9) je operace dilatace následovaná erozí. Otevření (8) je provedeno jako eroze po níž následuje dilatace. Otevření rozděluje plochy spojené tenkými elementy a eliminuje malé objekty. Proti tomu uzavření zaceluje malé otvory a spojuje objekty ležící blízko sebe. Pro zacelení větších otvorů lze nejdříve provést vícekrát operaci dilatace po níž je proveden stejný počet erozí. Obdobné pravidlo platí i pro otevření.

$$V \circ M = (V \otimes M) \oplus M \tag{8}$$

$$V \bullet M = (V \oplus M) \otimes M \tag{9}$$

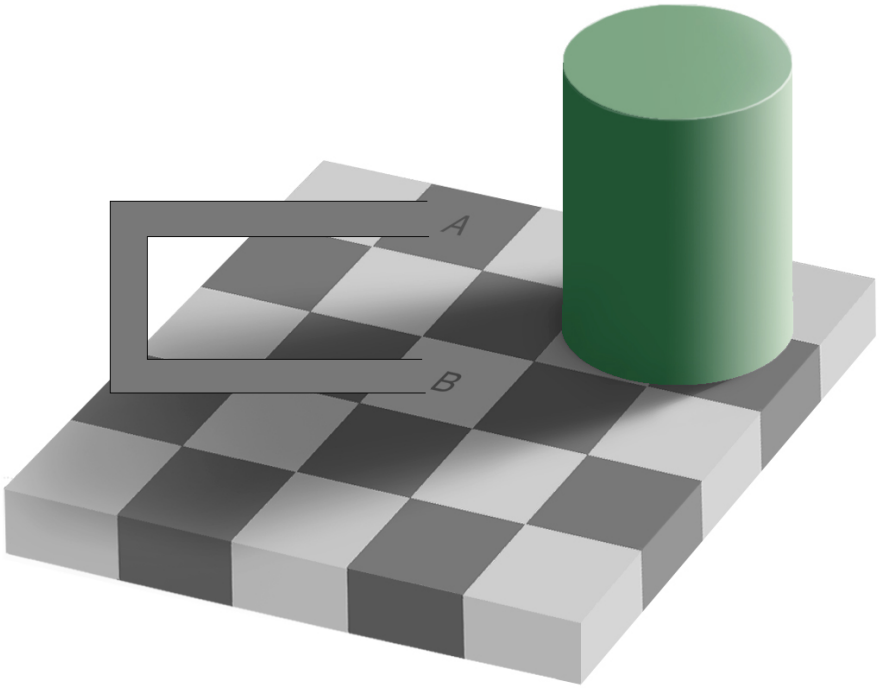
3 State of The Art

Tato část práce popisuje co je proces detekce objektů v obraze. Na začátků uvádím některá úskalí vidění a komplexitu celého procesu rozpoznání objektů. Následně nastiňuji možnosti využití počítačového vidění a jeho budoucnost. Závěrem kapitoly uvádím několik vybraných aplikací z praxe, využívajících detekci objektů či počítačové vidění a knihovnu pro práci s digitálním obrazem.

3.1 Detekování objektů v obraze

Detekce objektů v obraze je jedním z hlavních úkolů počítačového vidění. Při tomto úkolu se snažíme nalézt a identifikovat objekty v obraze či video-záznamu. Člověk samozřejmě dokáže snadno a intuitivně nalézt v obraze objekty a zařadit je. Proces rozpoznávání objektů se však skládá z mnoha úkonů a podílí se na něm oči i mozek, který data zpracovává. Přičemž nevyužívá jen aktuální obraz ale uplatňuje získané zkušenosti. Tento zjednodušený model je využíván i pro počítačové vidění. Kamera zde poskytuje obrazovou informaci. Ta je následně segmentována na zajímavé objekty, které se za využití strojového učení identifikují.

Ani člověk však nedokáže v obraze správně vyhodnotit všechny objekty za všech okolností. Stejně tak nelze očekávat absolutní úspěšnost od stroje. Člověk je navíc oproti počítači omezen tím, že nedokáže svou pozornost rozdělit na více objektů ve scéně současně. Také porovnání barev objektů, které neleží vedle sebe, bývá pro člověka problémové. Pokud čtenář nechce uvěřit nedokonalostem lidského zraku, demonstruji na obrázku 7, jak může být lidské vnímání barev zrádné.



Obrázek 7: Příklad chybného vnímání barev,zdroj původního obrazu [12]

Na obrázku 7 člověku jeho mozek ze zkušenosti říká, že bílá dlaždice B situovaná ve stínu, je světlejší než osvětlená tmavá dlaždice A. Přitom mají obě dlaždice totožný odstín šedé. V tomto případě tedy uplatnění informace z paměti mělo větší význam než obrazová informace zaslaná snímačem (okem). Lze tedy říci, že detekce objektů je proces díky němuž se extrahují z obrazové informace znalosti o objektech na snímaném místě. Detekování objektů v obraze může mít mnoho využití a těm se věnuji dále.

3.2 Účel detekce objektů

Samotné detekovaní objektů v obraze nebývá většinou finálním produktem. Získané informace o scéně bývají dále využity dalšími procesy k řízení jiných procesů. Taktéž mohou poskytovat uživateli nějaký druh služby. V obraze nemusí být objekty jen detekováný. Pokud máme k dispozici i video-sekvenci je možné provádět úkoly spojené s analýzou pohybu objektů. Tedy vytvářet sledovací systémy. Predikovat chování sledovaných objektů nebo naopak sledovat celou scénu a rekonstruovat ji ve 3D.

Analýza pohybu

Analýza pohybu může být ze svého principu aplikována pouze na série snímků. Protože pohyb zde představuje změnu pozice objektu za časový krok. Krok je dán časovým odstupem po sobě jdoucích snímků. Součástí toho procesu je zpravidla hledání korespondencí objektů mezi jednotlivými snímky. Protože snímaný obraz je průmětem z 3D prostoru na plochu, bývá problém se změnami tvaru objektu. Toto se děje, když je objekt zabírán z různých úhlů. Proto se většinou nehledá shoda celého objektu. Místo něj jsou zvoleny zajímavé body, které jsou snadno detekovatelné. Například se může jednat o body rohové. Sledováním těchto bodů lze získat informace o pohybu tělesa.

Celou analýzu pohybu objektů ve scéně můžeme samozřejmě vztáhnout i na kameru. V tomto případě je potřeba vybrat několik statických bodů. Sledováním jejich pohybu v obraze se reverzně určuje pohyb kamery. Taktéž je možné se pokusit o rekonstrukci sledovaného tělesa pokud je znám pohyb kamery a předmět je statický, případně je-li kamera statická a sledovaný předmět je v záznamu pozorovatelný z více úhlů. Pro rekonstrukci okolí je však vhodnější použít metody záznamu jejichž výsledkem jsou přímo body ve 3D prostoru. Například laserová měření při 3D skenování (LIDAR).

Rozpoznávání objektů

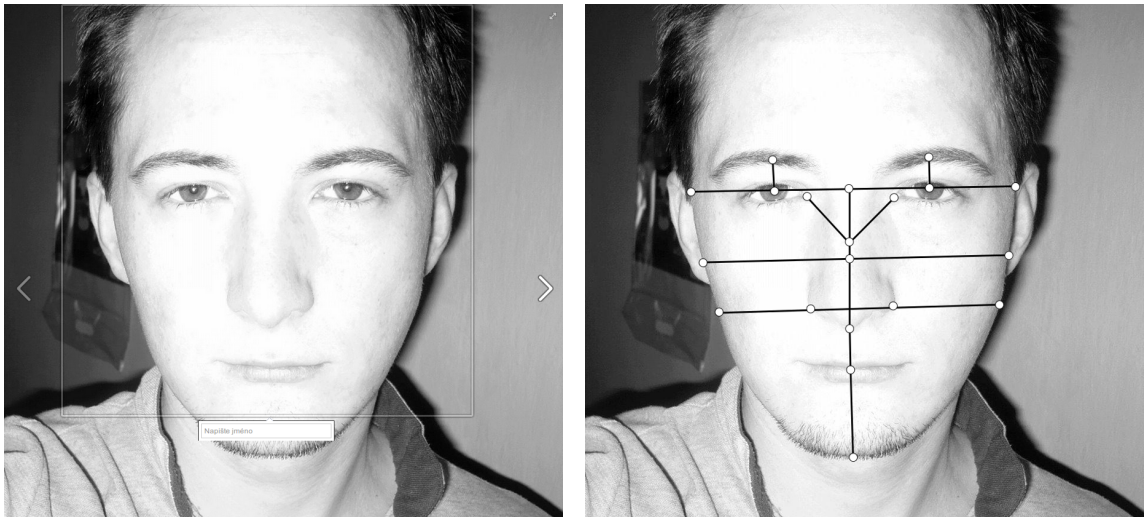
Při rozpoznávání objektů se snažíme najít v obraze očekávané entity z reálného světa, například obličeje osob, SPZ automobilů a podobně. Požadavky z praxe také vyžadují opačný přístup. Tedy situaci při níž ve scéně některé typy objektů nechceme. Například automobily v zóně zákazu vjezdu, osoby v nebezpečných či střežených prostorech a další případy. Při tomto problému je žádoucí detekovat pozadí a pak indikovat výskyt neznámého objektu ve scéně.

3.3 Příklady aplikace detekce objektů

V této sekci jsem uvedl dva příklady detekce objektů. Prvním je detekce obličejů. Tuto službu zná mnoho uživatelů internetu, protože je podporována mnoha servery na níchž můžete sdílet své fotografie s přáteli. Druhou aplikaci jsem zvolil z prostředí dopravy, aby byla blízko tématu této práce. Jedná se o detekci okrajů vozovky. Tato aplikace pracuje s videozáznamem kamery umístěné ve vozidle. V reálném čase zpracovává získaný digitální obraz.

3.3.1 Detekce obličejů na sociálních sítích

Za nejznámější příklad detekce objektů lze dnes považovat detekci obličejů. Tu svým uživatelům nabízí většina sociálních sítí. Což je velmi jednoduchý a chytrý způsob jak přiřadit ke snímkům dodatečnou informaci, k čemuž dopomáhají sami uživatelé označováním osob. Zohledněním těchto i dalších informací pak může být v sociálních sítích prováděna analýza síly sociálních vazeb mezi jednotlivými členy. Příklad detekce obličeje je uveden na obrázku 8a.



(a) Označování osob (b) Měření biometrických údajů

Obrázek 8: Příklad detekce obličejů

Z obrázku 8a je zjevné, že aplikace se při hledání obličeje omezuje na klíčové složky jako ústa, nos a oči. Za nedostatek považuji oddělení brady ze zvýrazněné oblasti. Oproti měření biometrie by měl být při označování osob na fotografiích kladen důraz na estetiku. Obličej by tedy měl být vybrán celý. Chválím však zahrnutí uší, které jsou zřejmě přidány jako nepovinný prvek. Naproti tomu při měření biometrických údajů (obr. 8) musí být odebírání vzorků normované a jednotné, aby bylo možné obličeje porovnávat [11].

3.3.2 Detekce okrajů vozovky

Tento projekt vypracoval Martin Gluč, který jej také prezentuje ve své práci [3]. Cílem tohoto projektu je detekovat okraje vozovky v městském prostředí. Zde se předpokládají lepší podmínky viditelnosti a udržovaného značení na silnici. Detektor v reálném čase zpracovává obraz z kamery umístěné ve vozidle a jako výstup poskytuje polohu nalezených okrajů vozovky v obraze.

Obrázek 9: Detekce okrajů vozovky, zdroj [3, str. 34]

Obrázek 9 vizualizuje nalezené okraje jízdních pruhů na vozovce. Detekce okrajů vozovky je zde provedena za pomoci houghovy transformace. Z výsledků houghovy transformace jsou získány nejlépe vyhovující linie, přičemž je zohledněna i návaznost pozic okrajových čar vozovky mezi jednotlivými snímky videa.

3.4 Knihovna OpenCV

Pro práci s obrazem jsem v projektu detektoru přejezdových závor využil knihovnu OpenCV. Jedná se o multiplatformní open source knihovnu zaměřenou na zpracování digitálního obrazu a počítačové vidění. Knihovna je implementována v programovacím jazyce C++. OpenCV poskytuje také interface pro programovací jazyky Python, Java, C. V nejnovějších verzích je průběžně doplňována funkcemi pro paralelní operace prováděné na GPU. Přičemž dokáže využívat jak OpenCL tak CUDA. Mezi podporované operační systémy patří Windows, Linux i OS X. Knihovna nabízí dobrou dokumentaci a stejně tak dobře jsou okomentovány i zdrojové kódy. Na stránkách projektu lze též nalézt velké množství příkladů [13].

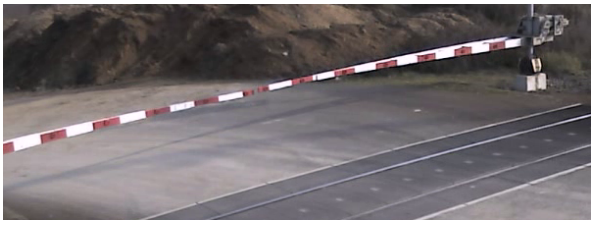
4 Detektor stavu přejezdových závor

Hlavní částí této práce je návrh a implementace nástroje pro autonomní kontrolu stavu přejezdových závor. Vyzkoušel jsem dva přístupy k problému. Prvním je metoda prahování s následnou klasifikací nalezených podezřelých objektů a vyhodnocením jejich vztahů. Druhá varianta využívá strojového učení. Konkrétně se jedná o metodu strojového učení SVM, která vyžaduje trénování na cvičných datech. Při tvorbě detektoru je pomocí vzorových příkladů trénováno SVM, které za ostrého běhu aplikace rozpoznává, zda se na vybraném místě nalézá či nenalézá přejezdová závora. Tento přístup vyžaduje ruční zadána pozice přejezdové závory pro každou kameru. Podrobně jsou oba přístupy rozebrány dále.

4.1 Detekce pomocí thresholdingu

Využití thresholdingu, tedy prahování byla první metoda, kterou jsem se pokusil závoru v obraze detekovat. Celý proces lze rozdělit na tyto kroky:

1. Thresholding pro červenou a bílou barvu
2. Úprava získaných binárních obrazů
3. Detekce objektů v binárních obrazech pomocí algoritmu connected component labeling
4. Výpočet příznaků pro jednotlivé objekty a selekce objektů
5. Hledání série objektů tvořících závoru



Obrázek 10: Závora s červenobílým pruhováním

V obraze jsem se snažil nalézt pixely náležící do červenobílého pruhování závor (obr. 10). Tento úkol jsem rozdělil na detekci bílých a červených objektů. Samotná detekce podezřelých objektů je provedena prahováním (2.2.2), které je prvním krokem procesu. Nejprve jsem se pokoušel o vhodnou volbu prahu v barevném modelu RGB. Vhodnějším se však ukázal převod do barevného modelu HSV. V obou případech je výsledkem prahování binární obraz, respektive obrazy. Protože prahování je prováděno pro červené a bílé objekty zvlášť.

Binární obrazy jsou před vyhodnocením ještě upraveny pomocí operací otevření a uzavření (viz 2.2.4). Tyto úpravy jsou nutné, protože závory bývají opatřeny červenobílým pruhováním pouze z vnějšku. Vnitřní strany zůstávají bílé a proto mohou pruhy splynout v jeden objekt, když je kamera zabírá z vyvýšené pozice. Po úpravách jsou v binárních obrazech vyhledávány oblasti náležící jednotlivým objektům.

Identifikace souvislých oblastí v získaných binárních obrazech je provedena metodou connected component labeling. Tuto metodu jsem zvolil, protože je jednoduchá pro implementaci,

ale účinná a pro účely tohoto detektoru postačující. Při hledání souvislé plochy objektu jsou taktéž zjišťovány některé příznaky, čímž je porušena modularita řešení, ale snížen potřebný počet průchodů obrazem.

Čtvrtým krokem je výpočet příznaků, které nebyly zjištěny během identifikace souvislých oblastí. Podle těchto příznaků jsou vybrány podezřelé objekty, které mohou být závorou. V tomto procesu se uplatňuje i vzájemná poloha objektů. Konečné rozhodnutí o přítomnosti přejezdové bariéry je však provedeno až v závěrečném kroku.

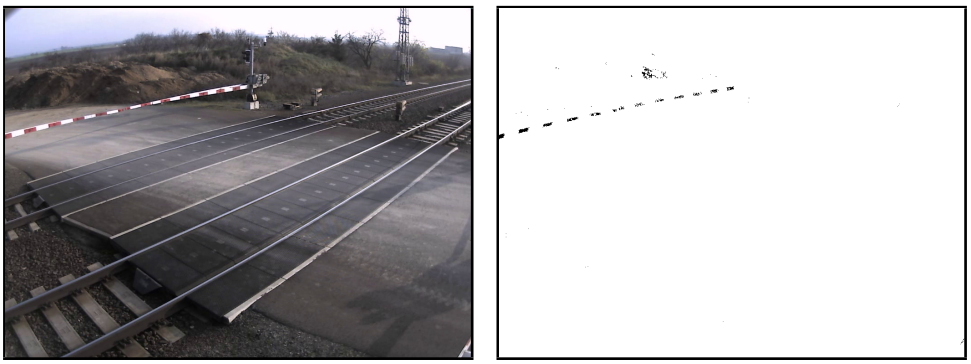
V závěrečném kroku je v množině podezřelých objektů hledána série objektů, které splňují podmínky kladeny na závoru. Pokud není takováto série objektů nalezena, je výsledkem detekce, že se závora ve snímku nenachází. V opačném případě je výskyt závory potvrzen.

4.1.1 Prahování

V obraze jsem se snažil nalézt pixely, které představují závoru. Za podezřelé mohu pixely označit podle jejich barevného odstínu. Přičemž využívám metody prahování (viz sekce 2.2.2). Při tom vycházím z toho, že závora je opatřena červenobílým pruhováním. Hledám tedy červené a bílé objekty. Při prahování jsem zkoušel používat dva barevné modely: RBG, HSV.

RBG prahování

Prahování na základě barevného modelu RGB bylo prvním pokusem o segmentaci obrazu. Protože v obraze hledám prahováním červené a bílé objekty je nejprve nutné definovat tyto barvy v RGB modelu. Pro kanály musí platit, že převažuje červený kanál. Tedy $R > G$ a $R > B$. Dále by hodnota kanálu R měla být alespoň nad polovinou rozsahu hodnot kanálu. Pokud tedy uvažuji celočíselné hodnoty v rozsahu odpovídajícím 1 byte, měla by být hodnota červeného kanálu $R > 126$.



(a) Vstupní obraz (b) Výsledek prahování

Obrázek 11: Příklad RGB prahování pro červené odstíny

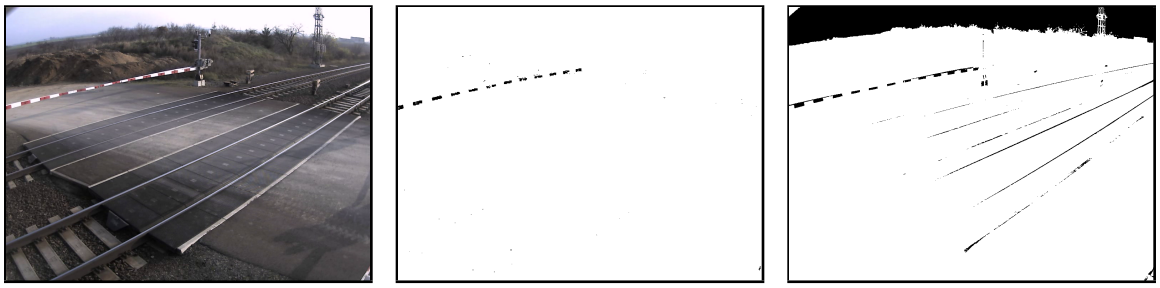
Příklad prahování v modelu RGB pro červený barevný odstín (obr. 11). V tomto příkladu byl zvolen práh tak jak je popsáno výše. Proces prahování byl proveden nad vstupním obrazem (obr. 11a). Na něm je zachycena přejezdová závora, za kterou se nachází půdní zavážka. Výsledný binární obraz (obr. 11b) nabývá jednotkových hodnot pro pixely, které splňují podmínku prahování. Zbylé pixely jsou nulové. Nulové pixely jsou vizualizovány bílou barvou a jednotkové pixely černou. Červené pruhy na závoře byly označeny správně. Spolu s

nimi však byly označeny i některé hnědé odstíny z navážky a vytvořili významně velkou plochu. Toto je způsobeno volbou prahu, která je v barevném modelu RGB pro vnímání člověka nepřirozená.

Jak je z příkladu i popisu volby prahu pro červené objekty patrné, není práce s RGB modelem jednoduchá a intuitivní. Což je způsobeno tím, že barevný odstín je závislý na hodnotě všech barevných kanálů. Výhodnějším se tedy jeví přejít do barevného modelu, který odstín barvy nastavuje pouze jedním kanálem. Takovým barevným modelem je HSV.

HSV prahování

Po převedení snímku do barevného modelu HSV je snadné stanovit prahové hodnoty barev za pomoci vzorníku, či palety. Nemohu zde však volit jedinou hodnotu prahu, ale interval ve kterém se budou vyskytovat barvy přijímaných pixelů. Barevný odstín se volí v kanálu H. V něm tedy vybírám červenou barvu. Složky S a V reprezentují sytost a jas. Protože nízké hodnoty sytosti vedou na obraz ve stupních šedi, nevolím dolní mez shodnou s minimální možnou hodnotou složky S. Pro nízkou hodnotu jasové složky V se výsledné barvy blíží černé. Proto i zde nevolím dolní mez shodnou s minimální hodnotou z rozsahu V. Experimentálně jsem stanovil meze intervalu pro červenou barvu v modelu HSV v OpenCV takto: $< (160, 50, 50); (179, 255, 255) >$. Tento interval jsem aplikoval na stejný vstupní obraz jako v příkladu (obr. 11). Obdobně jsem postupoval i pro bílou barvu a stanovil tento interval: $< (0, 0, 200); (179, 65, 255) >$.



(a) Vstupní obraz (b) Prahováno pro červenou barvu (c) Prahováno pro bílou barvu

Obrázek 12: Příklad HSV prahování

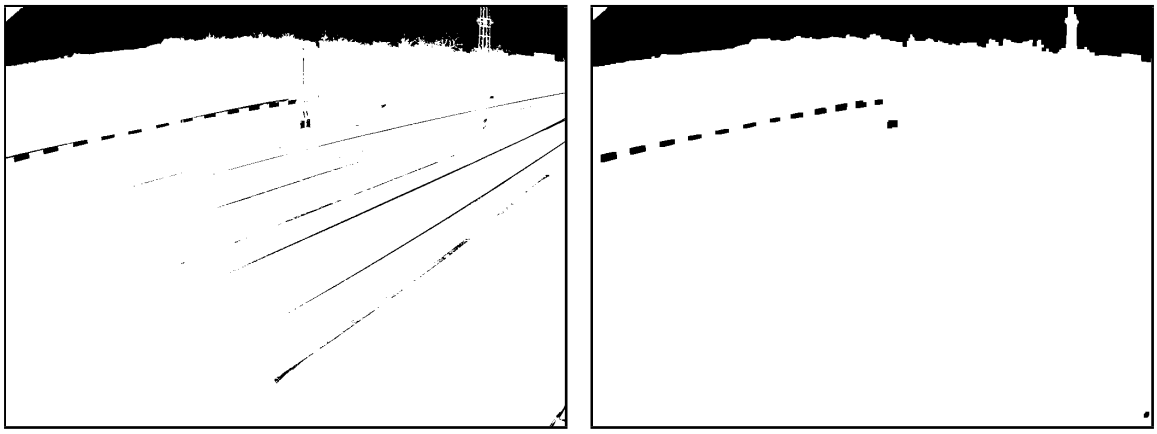
Příklad prahování v barevném prostoru HSV pro výše stanovený interval červené barvy je na obrázku 12. Vstupní obraz (obr. 12a) obsahuje přejezdovou závoru, za kterou se nachází půdní navážka. Nad vstupním obrazem je proveden proces prahování v daném HSV intervalu. Výsledkem je binární obraz (obr. 12b) nabývající jednotkových hodnot pro pixely, které splňují podmínku prahování. Zbylé pixely jsou nulové. Nulové pixely jsou vizualizovány bílou barvou a jednotkové pixely černou barvou. Stejně jako v předchozím příkladu (obr. 11). Ve výsledném obrazu jsou jasně vybrány červené objekty. Pruhy přejezdových závor mají výrazně větší plochu než osamělé falešně pozitivní objekty. Tudíž lze tyto falešně pozitivní objekty snadno odfiltrovat. HSV prahování se tedy projevilo jako účinější.

Prahováním jsem určil zda barvy jednotlivých pixelů spadají do stanovených intervalů. Tím jsem získal jeden binární obraz označující pixely náležící bílým objektů (obr. 12c) a druhý binární obraz pro pixely červených objektů (obr. 11). Se vzniklými binárními obrazy dále pracuji pomocí nástrojů matematické morfologie.

4.1.2 Úprava binárních obrazů

Binární obrazy jsou výsledkem prahování a identifikují pixely se, kterými se bude dále pracovat při vyšetřování oblastí náležících objektům. Proces prahování však může vybrat jen část objektu, nebo může objekt rozdělit na několik částí. Také se stává, že některé z objektů budou propojeny. To vše může být například zapříčiněno osvětlením a stíny ve scéně či ztrátou části informace o barvách vlivem komprese obrazu. Proto je potřeba binární obrazy před dalším zpracováním upravit. Zacelit přerušené objekty, vyplnit chybně detekované otvory v objektech a rozdělit chybně spojené objekty. Za tímto účelem využívám operací matematické morfologie (viz 2.2.4).

Pro přerušení tenkých spojení mezi objekty a eliminaci malých chybně detekovaných oblastí, jako jsou například odlesky, využívám operace *eroze*. Poté následuje několikanásobná operace *dilatace*. Jedná se tedy o operaci *otevření*. V pruzích přejezdové závory vznikají při thresholdingu otvory a trhliny, protože barva některých pixelů závory je ovlivněna ztrátovou grafickou kompresí obrazu při ukládání. Proto provádím na závěr operaci *uzavření*.



(a) Vstupní binární obraz (b) Výsledný binární obraz

Obrázek 13: Úpravy pomocí matematické morfologie

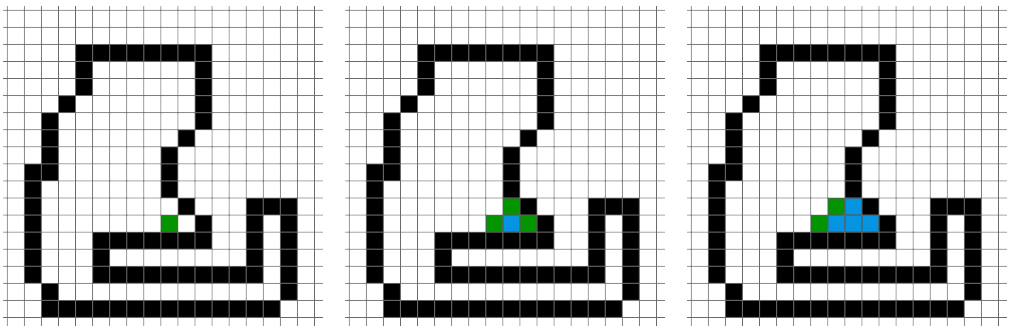
Příklad (obr. 13) ukazuje praktický význam operací matematické morfologie. Vstupním obrazem (obr. 13a) je snímek prahovaný pro bílou barvu. Ve vstupním obraze jsou některé z nalezených pruhů závor spojeny tenkými přechody. Stalo se to, protože červenobílé pruhování je na závorách pouze z vnějších stran a uvnitř jsou závory bílé. Takže kamera snímající přejezd z výšky vidí částečně i vnitřní stranu závor, což pruhy spojí. Taktéž odlesky kolejí tvoří dlouhé tenké lineární oblasti. Aplikací výše uvedeného sledu operací vzniká upravený binární obraz (obr. 13b). V něm jsou pruhy na závoře osamostatněny. Odstraněny jsou drobné odlesky a tenké linie kolejí. V obraze se zmenšilo množství souvislých ploch, což při další práci zjednoduší jeho prohledávání a sníží množství podezřelých objektů.

Když jsem získal vhodný binární obraz začnu v něm hledat souvislé plochy, které mohu označit jako objekty. Jedná se tedy o spojování jednotlivých pixelů v plochy. To provádím pomocí metody connected component labeling.

4.1.3 Connected component labeling

Tento algoritmus vychází z teorie grafů a slouží k hledání komponent grafu. V počítačovém vidění se používá pro detekci souvislých oblastí v binárních obrazech. Při práci s obrazem je na pixely nahlíženo jako na uzly grafu. Přičemž dva pixely jsou spojeny hranou pokud se sebou přímo sousedí. Při tvorbě tohoto detektoru považuji za sousedící pixely ty, které mají společnou hranu. Pro zaznamenání příslušnosti pixelu k nějaké oblasti jsou použity indexy (štítky). Ty jsou vepisovány do pomocného obrazu. Všechny pixely jedné spojité oblasti jsou označeny stejným indexem. V pomocném (výstupním) obraze je tedy ve výsledku tolik unikátních indexů, kolik je v obraze nalezeno oblastí plus jeden "nulový"index pro označení pozadí. Postupně je procházen celý binární obraz. Pro každý pixel je vyvolán algoritmus hledání souvislé komponenty. Tyto i dále uvedené informace o algoritmu connected component labeling jsem získal z [4] a [2, str. 69 - 73].

Při zpracování jednoho pixelu kontroluje algoritmus zda není pixel v binárním obraze označen jako pozadí a zda již není ve výstupním obraze indexován jiným objektem. Pokud jsou podmínky splněny je pixel indexován jako součást aktuálně vznikajícího objektu a jsou rekurzivně zkoumány i pixely na sever, východ, jih a západ od aktuální pozice. Pokud podmínky splněny nejsou zpracování aktuálního pixelu končí.



(a) První iterace (b) Druhá iterace (c) Třetí iterace

Obrázek 14: Znázornění postupu algoritmu

Příklad (obr. 14) znázorňuje sekvenční indexaci pixelů v obraze. Zelenou barvou jsou vyznačeny pixely zpracovávané v aktuální iteraci. Modrou barvou jsou označeny pixely, které algoritmus přidělil ploše objektu. Bílá barva náleží pixelům, které jsou objektem a zatím nebyly nikam přiřazeny. Černá barva říká, že pixel náleží pozadí. V prvním kroku (obr. 14a) je vybrán počáteční pixel. V druhém kroku (obr. 14b) je tento pixel zpracován a přidělen objektu. Pixely severně, východně a západně jsou v aktuální iteraci zpracovávány. Přičemž jižní pixel tvoří hranici objektu a proto nebude indexován. Ve třetí iteraci (obr. 14c) je zobrazeno, jak celý proces rekurzivně pokračuje.

Když jsou prozkoumány všechny podezřelé pixely algoritmus kočí. V indexovaném (výsledném) obraze je uložena informace o tom, kterému objektu jednotlivé pixely náleží. Přičemž speciální hodnota indexu určuje pixely pozadí. Pro zacházení s nalezenými objekty slouží v detektoru závor třída *ImgObject*. Dále je potřeba objekty kvalitativně porovnat, roztřídit a vybrat z nich pouze elementy jenž tvoří závoru. Proto určuji pro každý nalezený objekt číselně porovnatelné atributy.

Atributy objektů zjišťované během indexace

Obvod a obsah objektu lze určit už při indexaci daného objektu, čímž snižuji počet průchodů obrazem. Obsah objektu je roven součtu pixelů náležících do objektu. Prakticky je implementován jako čítač indexovaných pixelů. V detektoru přejezdových závor je označen jako atribut *myArea* náležící třídě *ImgObject*. Obvod objektu lze spočítat při indexaci jako součet událostí, kdy sousedící pixel nebyl přiřazen do aktuálně vytvářeného objektu. Výpočet obvodu tedy přidává do algoritmu jeden čítač. V detektoru přejezdových závor představuje obvod atribut *myPerimeter* náležící třídě *ImgObject*.

4.1.4 Selekce objektů

Když jsou v obraze nalezeny objekty odpovídající souvislým plochám v binárním obraze, je třeba ze vzniklé množiny vybrat jen ty objekty, které mohou náležet přejezdové závoře. Proto se objektům přiřazují porovnatelné atributy. Souhrnně se takové atributy nazývají příznaky. Detektor přejezdových závor využívá jako příznaky obvod, obsah, minimal bounding box a některé z nich odvozené příznaky. Získávání obvodu a obsahu jsem již popisoval výše.

Minimal boundin box

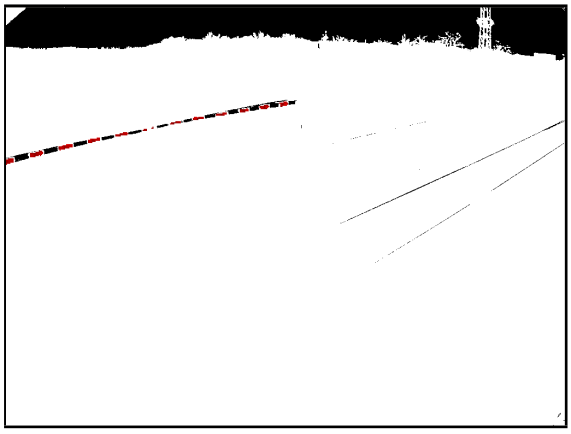
Nejmenší obalující obdélník je struktura mající nejmenší obsah ze všech obdélníků, které obsahují všechny pixely daného objektu či skupiny objektů. V tomto textu uvažuji a využívám pouze bounding boxy zarovnané s osami. Bývají též označovány jako axis aligned bounding box (AABB)[5]. AABB zjednodušuje operaci průniku, kterou používám při práci s nalezenými skupinami objektů. V detektoru závor je struktura reprezentující bounding box implementována pod názvem *AABB*.

Při selekci podezřelých objektů, které mohou být součástí závory jsou vyřazeny objekty mající zanedbatelnou plochu, tedy jsou velmi malé. Taktéž vyřazuji objekty vyplňující méně než polovinu plochy svého AABB. Toto opatření odfiltruje podlouhlé tenčí objekty, jejichž směr není rovnoběžný s horizontální nebo vertikální osou kamery. To jsou například koleje či kabely, které díky perspektivě a jejich délce nebývají vodorovné s horizontální osou kamery.

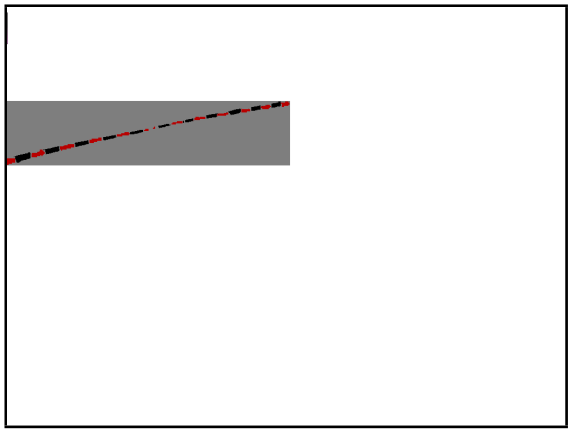
Oblast zájmu

Ke zmenšení množiny podezřelých objektů lze využít toho, že se na závoře musí vyskytovat bílá i červená barva současně. Tudíž mne v obraze nezajímají oblasti, ve kterých se vyskytuje jen jeden ze dvou hledaných barevných objektů. Nezajímavé oblasti mohu vyloučit jednoduchým postupem. Vytvořím dva velké bounding boxy. Jeden pro červené objekty a druhý pro bílé. Nyní vyberu ze všech objektů ve scéně ty, které leží celou svou plochou současně v AABB pro červené objekty a AABB pro bílé objekty.

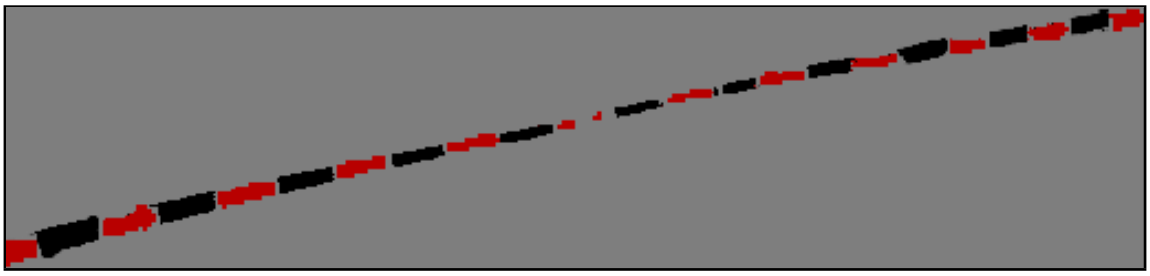
Příklad (obr. 15) ukazuje jak může oblast zájmu omezit množství podezřelých objektů. Vstupní obraz (obr. 15a) obsahuje všechny červené (vyznačeny červenou barvou) i bílé objekty (vynačeny černou barvou) detekované ve scéně. Bílé objekty jsou roztroušeny po celé ploše snímku. Červenými objekty jsou jen pruhy závor, které jsou koncentrovány na omezené ploše. Vytvořením průniku AABB bílých a červených objektů vzniká AABB v podstatě odpovídající červenému AABB. Oblast průniku je vyznačena na obrázku 15b šedou barvou. Vybrané jsou pak pouze objekty ležící uvnitř této oblasti. Při porovnání plochy obrázku a průniku AABB je jasné, že dochází k zefektivnění další práce se snímkem.



(a) Ojekty v obraze



(b) Oblast zájmu



(c) Detail oblasti zájmu

Obrázek 15: Selekce oblasti zájmu

Selekce pomocí oblasti zájmu nemá za cíl označit jen elementy závory. Spíše odstraní objekty jenž závorou být nemohou. Jedná se tedy o filtr. Úspěšně odstraňuje například velké bílé oblasti oblohy. Po odfiltrování některých nezajímavých objetů je na čase ve zbylé množině podezřelých objektů hledat závoru.

4.1.5 Nalezení závory

Při hledání závory v obraze je důležité vybrat a popsat podmínky, které mají elementy, detekované objekty jenž tvoří závoru, splňovat.

- elementy tvořící závoru leží na jedné přímce
- vedle sebe nesmí ležet dva elementy stejné barvy
- série elementů musí být tvořena alespoň pěti členy

Pro vyhledání závory jsem se inspiroval Jarvisovým algoritmem pro konstrukci konvexní obálky [9]. Při implementaci jsem postupoval takto: podezřelé červené a bíle objekty jsou spojeny do jedné množiny. Množina je setříděna podle horizontální osy obrazu. Podezřelé objekty jsou procházeny přičemž se kontroluje střídání barev. Pro každé tři po sobě jdoucí

objekty je vypočten úhel svíraný středy jejich AABB. Svíraný úhel nesmí přesáhnout experimentálně zvolenou odchylku 0.27 *rad*. Místo středu objektu využívám střed AABB, protože v dřívějším postupu je ověřeno, že objekt tvoří minimálně polovinu plochy bounding boxu. Pokud je nalezena série pěti objektů splňujících tyto požadavky, je výsledným stavem nalezení závory. Pak jsou středy AABB počátečního a koncového objektu označeny za krajní body úsečky označující pozici závory v obraze. Jinak je výsledným stavem, že závora nalezena nebyla.

Obrázek 16: Vizualizace hledání závory

Na snímku z přejezdové kamery (obr. 16) jsou vizualizovány entity procesu hledání přejezdové závory v podezřelých objektech. Modrými obdélníky jsou zvýrazněny nalezené podezřelé objekty. Lomená čára barvy cyan, která je spojuje, představuje jejich horizontální seřazení při zpracování. Taktéž zobrazuje úhly v jakých na sebe objekty navazují. A konečně silná zelená úsečka označuje nalezenou pozici závory v obraze. V levém dolním rohu je navíc slovně popsán stav závory.

Závěrem ještě popíši návratové hodnoty detektoru přejezdových závor provedeného pomocí thresholdingu. Program vždy vrací alespoň jednu hodnotu. Ta indikuje zda byla nebo nebyla závora nalezena. V případě že závora nalezena byla jsou v dalších čtyřech hodnotách vráceny souřadnice koncových bodů úsečky označující pozici závory.

1. byla závora nalezena? (1-Ano | 0-Ne)
2. (závora nalezena) souřadnice řádku počátečního bodu závory
3. (závora nalezena) souřadnice sloupce počátečního bodu závory
4. (závora nalezena) souřadnice řádku koncového bodu závory
5. (závora nalezena) souřadnice sloupce koncového bodu závory

Tento detektor je dále otestován v kapitole experimentů. Také je porovnán s druhou verzí detektoru, která využívá strojového učení formou SVM.

4.2 Detekce pomocí SVM

Druhá metoda vytvořená k detekování železničních přejezdových závor využívá strojové učení. Zvoleným modelem učení je SVM, který je implementován v knihovně OpenCV. Tento detektor pracuje s obrazem ve stupních šedi. Takže je méně náchylný na změny světelných podmínek, protože kamera se při slabém osvětlení dokáže automaticky přepnout z RGB do IR režimu, který poskytuje obrazy pouze ve stupních šedi. Pro vytvoření a správnou funkčnost detektoru je potřeba provést tyto kroky:

1. Tvorba vzorů pro učení
2. Učení SVM
3. Klasifikace neznámých vzorů pro detekování závory

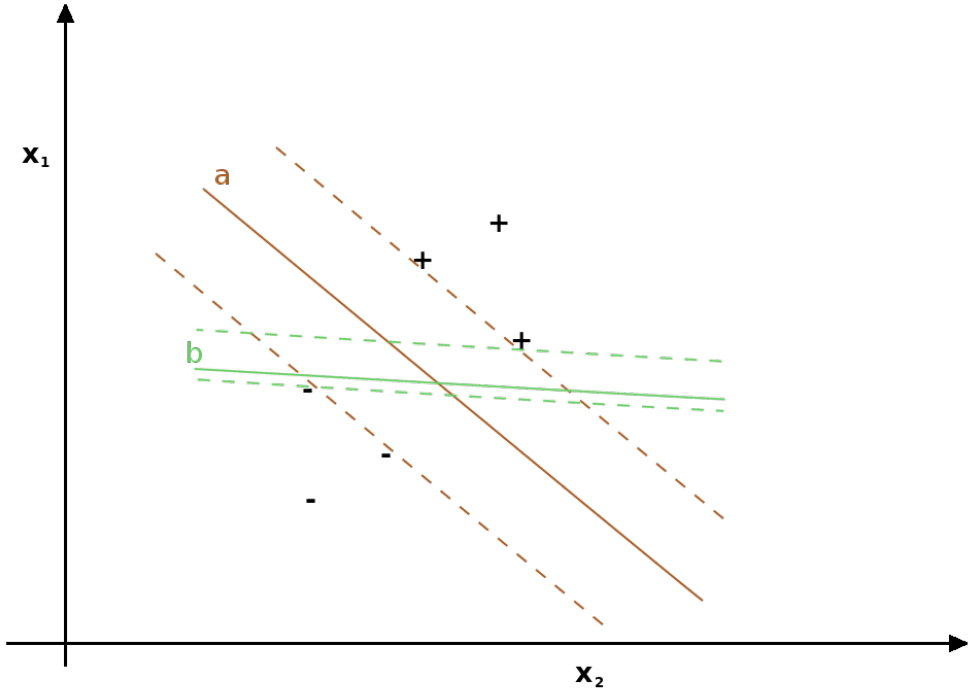
SVM je model strojového učení vyžadující učitele, tedy vzorové příklady. Proto je nutné vzorové příklady vytvořit a ručně klasifikovat. Je třeba vybrat jak příklady náležící oblasti závory, tak příklady které nemají být klasifikovány jako závora. Vzorky jsou odebírány z obrazu ve stupních šedi. Vzorky jsou normalizovány kvůli vlivu světelných podmínek a filtrovány pro odstranění šumu. Pozitivní a negativní vzorky jsou na závěr spojeny do jedné množiny trénovacích příkladů.

Při učení jsou SVM předloženy připravené příklady obsahující vektor příznaků a jeho klasifikaci. SVM pak vyhledá vzory pro předložené příklady a přizpůsobí jim svůj vnitřní stav. SVM je následně serializován a uložen pro budoucí použití při klasifikaci nezařazených vektorů příznaků.

Před samotnou klasifikací nových vzorků je potřeba pro každou kameru ručně vybrat pozici spuštěné přejezdové závory v obraze. Dále už detektor sám z vybrané oblasti závory odebírá vzorek. Na jeho základě je vytvořen vektor příznaků, obdobně jako při vytváření příkladů pro učení. Vzorek je pak klasifikován pomocí SVM a je rozhodnuto o přítomnosti přejezdové závory. Než však přistoupím k podrobnějšímu popisu detektoru věnuji se osvětlení principu strojového učení formou SVM.

4.2.1 Support vector machine

Support vector machine je metodou strojového učení. Teorii o SVM jsem čerpal z [8]. Aby mohl SVM správně pracovat, vyžaduje sadu trénovacích dat. Jedná se tedy o model učení s učitelem. Při učení s učitelem jsou stroji předkládány příklady, které tvoří dvojice složené z vektoru příznaků a požadovaného výstupu. Aby v budoucnu mohl stroj samostatně rozpoznávat další vektory, zobecňuje vhodným způsobem předkládaná data. SVM ke klasifikaci předkládaných vektorů příznaků využívá jednoduché myšlenky dělení prostoru.



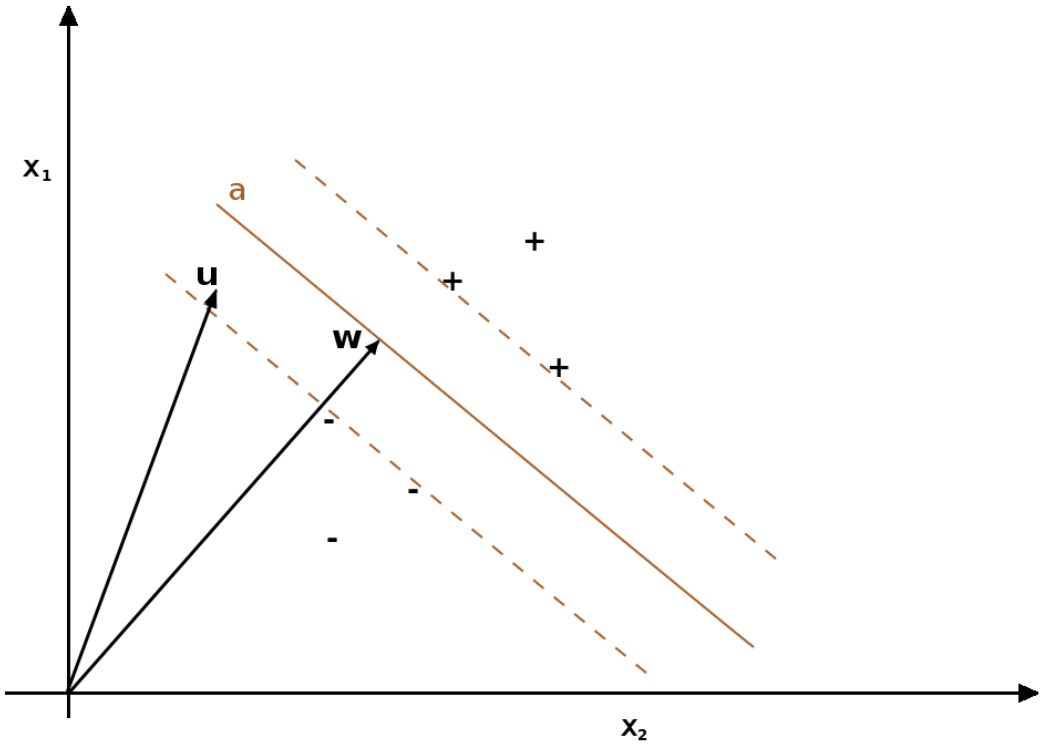
Obrázek 17: Problém dělení prostoru

Ve své základní verzi dělí SVM prostor příznaku na dva podprostory. V příkladu (obr. 17) jsou sledovány dva příznaky x_1 a x_2. Prostor je tedy dvou dimenzionální. Tento prostor obsahuje šest vzorků. Stroj se snaží zjistit, jak prostor rozdělit, aby každá polorovina obsahovala pouze jeden typ vzorků. Pozitivní vzorky jsou označeny symbolem $+$. Negativní zase symbolem $-$. Nejjednodušším způsobem rozdělení této roviny je přímka. Ovšem přímek, jenž od sebe oddělují skupiny pozitivních a negativních příkladů je nekonečně mnoho. V příkladu jsou zobrazeny dvě: zelená přímka a a hnědá přímka b. Ty se od sebe liší tím, v jaké vzdálenosti prochází od nejbližších bodů z obou skupin. Tato vzdálenost je znázorněna čárkovanými čarami. Problémem je výběr nejlepší přímky ze všech možných.

K porovnání kvality přímek lze využít šířky pruhu, který je dán vzdálenostmi k nejbližším bodům z obou skupin příkladů. Hledán je tedy pruh s maximální šířkou, který dělí prostor příznaků způsobem popsaným výše. Intuitivně je pak jasné, že nejvhodnější přímka z daného pruhu má stejně daleko k oběma okrajům pruhu s nimiž je rovnoběžná. Z příkladu (obr. 17) je nejširší pruhem ten s hnědými okraji. Nejvhodnější dělící přímka příkladu je tedy přímka a.

Klasifikace nových vzorků

Lze vypozorovat, že všechny vzorové příklady nejsou stejně významné pro nalezení rozdělení prostoru příznaku na poloroviny. Hlavními vzorky jsou vektory příznaků, které se nalézají na hranici pruhu vyznačené čárkovanými čarami (obr. 18). Odtud pochází název metody support vector machine.



Obrázek 18: Klasifikace vzorků

Obrázek 18 nastiňuje myšlenku stojící za postupem klasifikace nových vzorků. Přímka a dělí podprostor na pozitivní a negativní vzorky. Vektor $\mathbf{u}$ představuje nový vzorek o kterém se má rozhodnout zda je či není pozitivní. Vektor $\mathbf{w}$ je kolmý na přímku a. Úvaha klasifikace je taková: pokud je skalární součin vektorů $\mathbf{w}$ a $\mathbf{u}$ větší než vzdálenost k dělící přímce a, spadá $\mathbf{u}$ do poloroviny pozitivních vzorků. Jinak je negativní. Podmínku pozitivity lze tedy zapsat výrazem (10).

$$\mathbf{w} \cdot \mathbf{u} \geqq c \tag{10}$$

Převedením konstanty c na levou stranu a přeznačením dostávám nový tvar podmínky pozitivity.

$$\mathbf{w} \cdot \mathbf{u} + b \geqq 0 \quad \text{kde} \quad b = -c \tag{11}$$

Místo neznámého vstupního vektoru $\mathbf{u}$ mohu zvolit pozitivní vzorek (12) nebo negativní vzorek (13).

$$\mathbf{w} \cdot \mathbf{x}_+ + b \geqq 1 \tag{12}$$
$$\mathbf{w} \cdot \mathbf{x}_- + b \leqq -1 \tag{13}$$

Dále mohu zavést proměnou y_i definovanou takto:

$$y_i = \begin{cases} +1, & \text{pro pozitivní vzorky} \\ -1, & \text{pro negativní vzorky} \end{cases} \tag{14}$$

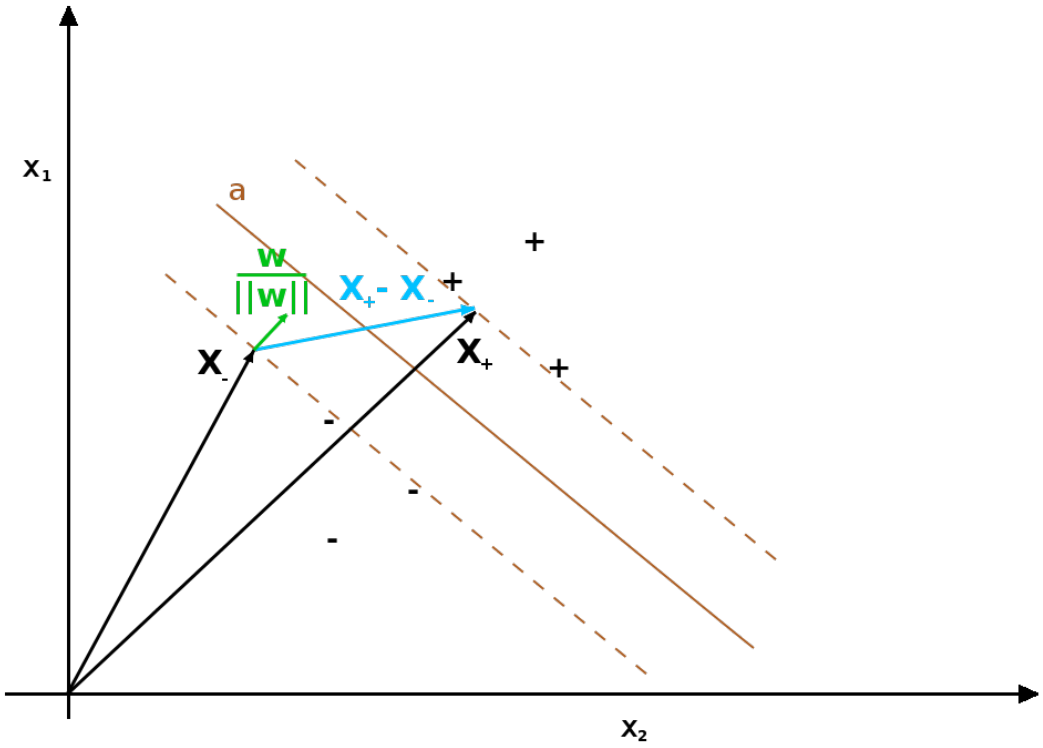
Pokud proměnou y_i vynásobím rovnice (12) a (13) dostanu dva stejné výrazy ve tvaru (15).

$$y_i(\mathbf{w} \cdot \mathbf{x_i}) + b - 1 \geqq 0 \tag{15}$$

Podmínky na šířku pruhu kladou jen vzorové vektory ležící na okraji pruhu. Nikoli v celé polorovině. Proto upravím nerovnici na rovnici.

$$y_i(\mathbf{w} \cdot \mathbf{x_i}) + b - 1 = 0 \quad \text{pro } x_i \text{ na okrajích pruhu} \tag{16}$$

Dále je potřeba zjistit vztah pro šířku pruhu.



Obrázek 19: Šířka pruhu

Obrázek 19 znázorňuje vztahy platící pro šířku pruhu. Vyberu ze vzorových příznakových vektorů po jednom z každé strany pruhu. Pro negativní stranu je to $\mathbf{x}_-$, pro pozitivní $\mathbf{x}_+$.

Jejich odečtením získám vektor $\mathbf{x}_+ - \mathbf{x}_-$ spojující obě strany pruhu. Tento vektor může být jakkoliv natočen a proto nelze jeho délku použít přímo pro zjištění šířky pruhu. Ale pokud znám jednotkový vektor kolmý na dělící přímku a, mohu za pomocí skalárního součinu tuto délku zjistit. Vektor $\mathbf{w}$ kolmý na a je definován výše. Není jednotkový a proto musí být normalizován. Čímž je získán výraz $\frac{\mathbf{w}}{\|\mathbf{w}\|}$. Šířka pruhu je tedy rovna výrazu (17).

$$width = (\mathbf{x}_+ - \mathbf{x}_-) \cdot \frac{\mathbf{w}}{\|\mathbf{w}\|} \tag{17}$$

$$width = (\mathbf{x}_+ \cdot \mathbf{w} - \mathbf{x}_- \cdot \mathbf{w}) \frac{1}{\|\mathbf{w}\|} \tag{18}$$

Z výrazu (16) pak mohu vyjádřit:

$$\text{pro } x_+\text{: } \mathbf{w} \cdot \mathbf{x}_+ = b - 1 \tag{19}$$

$$\text{pro } x_-\text{: } \mathbf{w} \cdot \mathbf{x}_- = -b - 1 \tag{20}$$

Dosazením výrazů (19) a (20) do (18) dostanu:

$$width = \frac{2}{\|\mathbf{w}\|} \tag{21}$$

Jak už bylo řečeno dříve, hledán je pruh s maximální šířkou (22). Při hledání maxima nemá konstanta vliv (23). Hledání maxima ve zlomku, kde proměnná je pouze ve jmenovateli lze převést na hledání minima čitatele (24).

$$max(\frac{2}{\|\mathbf{w}\|}) \tag{22}$$

$$\Rightarrow max(\frac{1}{\|\mathbf{w}\|}) \tag{23}$$

$$\Rightarrow min(\|\mathbf{w}\|) \tag{24}$$

Šířka pruhu se může pohybovat pouze v kladných číslech. Proto je možné hledat i minimum proměnné umocněnou na na druhou (25). Vynásobené konstantou při hledání minima zde výsledek neovlivní (26).

$$min((\|\mathbf{w}\|)^2) \tag{25}$$

$$\Rightarrow min(\frac{1}{2}(\|\mathbf{w}\|)^2) \tag{26}$$

K nalezení dělící přímky je tedy potřeba nalézt extrém z výrazu (26) a dodržet podmínku výrazu (16). K nalezení extrému za určitých podmínek jsou zavedeny Lagrangerovy multiplikátory α.

$$L = \frac{1}{2}(\|\mathbf{w}\|)^2 - \sum_{i=1}^{n} \alpha_i [y_i(\mathbf{w} \cdot \mathbf{x_i}) + b - 1] \tag{27}$$

Nyní je hledán extrém výrazu L (27). Jsou tedy provedeny derivace. Při hledání extrémů zjišťujeme, kdy jsou derivace rovny nule. Výsledkem pro derivaci podle $\mathbf{w}$ je výraz (28). Pro derivaci podle b dostanu výraz (29)

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x_i} \tag{28}$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{29}$$

Omega je tedy lineárně závislá na trénovacích datech. Zahrnutím předchozí úvahy o tom, že pruh ovlivňují pouze vzorové příklady na jeho hranici je jasné, že velké množství α_i bude nulových. Dále je možné dosadit za $\mathbf{w}$ do výrazu (27) a zjistit na čem je závislý vyhledávaný extrém.

$$L = \frac{1}{2}\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x_i} \cdot \mathbf{x_j} + b\sum_{i=1}^{n} \alpha_i y_i \tag{30}$$

$$L = \frac{1}{2}\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x_i} \cdot \mathbf{x_j} \tag{31}$$

Dosazením za $\mathbf{w}$ byl získán výraz (30). Na něj lze aplikovat (29), čímž je vynulován člen b. Výsledný výraz je (31). Vyhledávaný extrém je tedy závislý na skalárním součinu $\mathbf{x_i} \cdot \mathbf{x_j}$.

$$(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x_i}) \cdot \mathbf{u} + b \geqq 0 \tag{32}$$

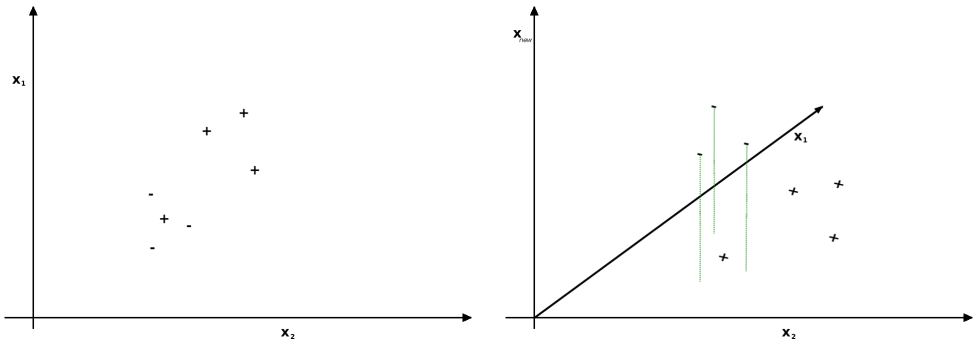
Když je znám předpis pro $\mathbf{w}$ (28) je možné jej dosadit do podmínky pozitivity (11), čímž vznikne výraz (32), kterým je možné klasifikovat neznámé vektory příznaků na základě trénovacích dat.

Separabilita dat

V některých případech se může stát, že cvičná množina příznakových vektorů není v daném prostoru příznaků lineárně rozdělitelná (obr. 20a). Myšlenkou jak tento problém vyřešit je: "Pokud si nevíte rady zkuste se na problém podívat z nové perspektivy". V případě prostoru příznakových vektorů se přidá nový rozměr. V něm už data mohou být separovatelná.

Příklad (obr. 20) zobrazuje, jak může změna pohledu vést k lineární separabilitě (obr. 20b) původně lineárně nerozdělitelných dat (obr. 20a). Ve dvou dimenzionálním prostoru se mohou zdát data lineárně nerozdělitelná. Přidáním dalšího rozměru vzniká hyperprostor, ve kterém mohou být data rozdělena pomocí hyperplochy. V příkladu vzniká trojrozměrný prostor. Ten může být rozdělen rovinou na dva podprostory.

Pro převod obecného vektoru příznaků $\mathbf{x}$ do nového prostoru lze definovat zobrazení $\phi(\mathbf{x})$. Ovšem jak vyplynulo z výrazu (31), je vyhledávání extrému závislé na skalárním součinu $\mathbf{x_i} \cdot \mathbf{x_j}$. Pro rozšířený prostor tedy platí: $\phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_j})$. Pro rozpoznání neznámého vektoru je taktéž



(a) Cvičná data (b) Přidaný rozměr

Obrázek 20: Změna perspektivy

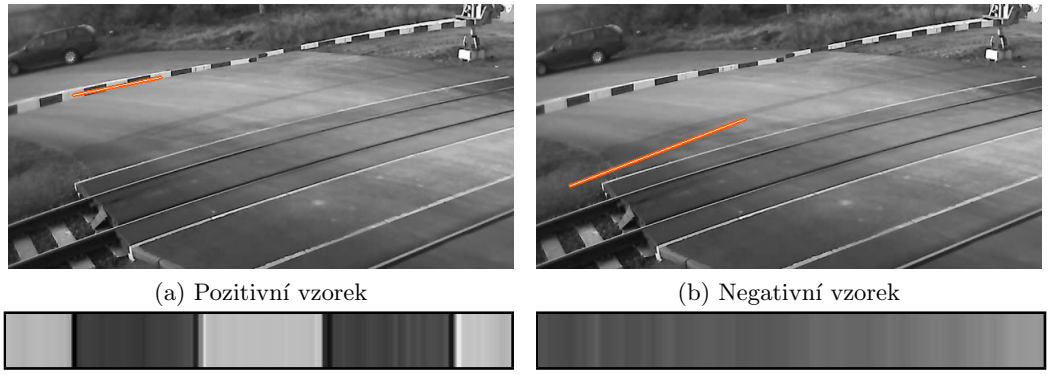
potřeba skalární součin $\mathbf{x_{sum}} \cdot \mathbf{u}$. Jeho zobrazení v rozšířené prostoru je $\phi(\mathbf{x_{sum}}) \cdot \phi(\mathbf{u})$. Je však vidět, že pro žádný z výpočtů není potřeba definovat zobrazení pro samostatný vektor $\mathbf{x}$. Stačí definovat funkci, která přiřadí dvojci vektorů hodnotu jejich skalárního součinu v rozšířeném prostoru (33).

$$k(\mathbf{v_1}, \mathbf{v_1}) = \phi(\mathbf{v_1}) \cdot \phi(\mathbf{v_2}) \tag{33}$$

Díky tomuto postupu je možné rozdělit původní prostor nelineárně, bez nutnosti definovat zobrazení. Soubor funkcí používaných jako $k(\mathbf{v_1}, \mathbf{v_1})$ se nazývá "kernel methods"do češtiny jádrové funkce.

4.2.2 Tvorba příkladů a trénování SVM

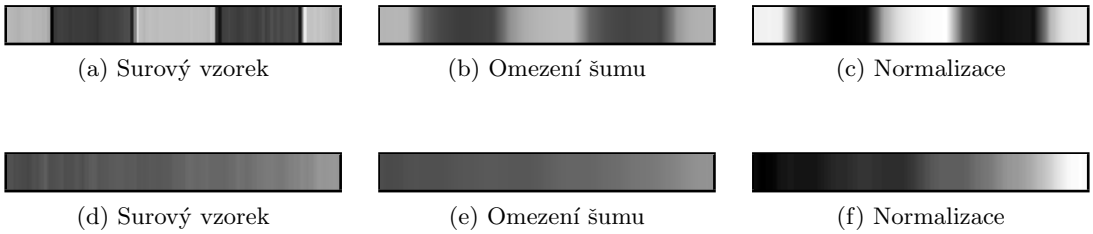
Pro správnou funkčnost detektoru je nutné nejprve naučit SVM, které vektory příznaků náleží závoře a které ne. Proto je nutné nejdříve vytvořit trénovací množinu příkladů. Příklad je tvořen dvojicí. Jedním členem je vektor příznaků a druhým je klasifikace vzorku. Hodnota $+1$ značí, že vzorek je hledaný objekt tj. pozitivní příklad. Hodnota -1 označuje negativní příklad.



(a) Pozitivní vzorek (b) Negativní vzorek

Obrázek 21: Výběr vzorků

Příklad (obr. 21) ukazuje odebírání vzorků z obrazu. Výběr je proveden pomocí zvýrazněné úsečky jejíž pozice byla zadána manuálně. Po délce úsečky je odebrán požadovaný počet vzorků barvy. Při odebírání je využito interpolace barvy. Tyto barevné vzorky jsou složkami jednoho příznakového vektoru. Délka úsečky může být tedy různá, ale počet odebraných hodnot je vždy stejný. Na obrázku vlevo (obr. 21a) je zachycen výběr pozitivního vzorku ze snímku a vizualizace zjištěného příznakového vektoru. Na obrázku vpravo (obr. 21b) je totožný úkon proveden pro negativní vzorek.



(a) Surový vzorek (b) Omezení šumu (c) Normalizace

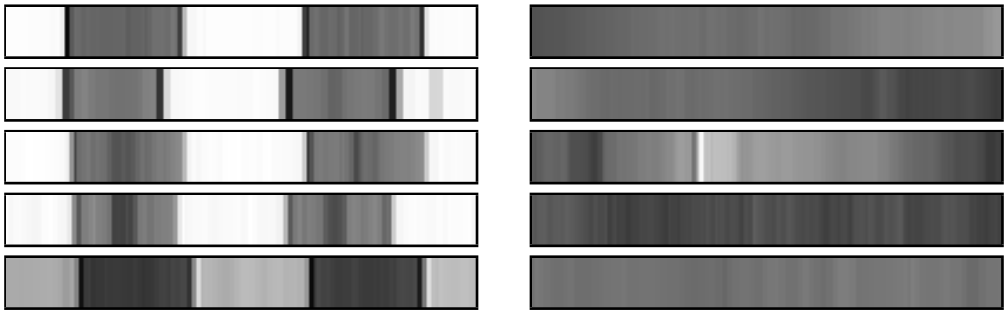
(d) Surový vzorek (e) Omezení šumu (f) Normalizace

Obrázek 22: Zpracování vzorku

Po odebrání surových vzorků je potřeba vzorky filtrovat pro odstranění šumu a vylepšení kontrastu. Příklad (obr. 22) zobrazuje postupnou filtraci surového vzorku (obr. 22a). Na vstupním vzorku je patrné, že barva červených a bílých pruhů závory není jednolitá. Tento problém je způsoben šumem ve zdrojovém obraze. K odstranění šumu se v obraze používá průměrování pomocí konvoluce (viz 2.1.2). V případě vektoru příznaků bude konvoluce prováděna na jednorozměrném poli místo dvourozměrného pole (obrazu). Výsledkem je vektor s omezeným vlivem šumu (obr. 22b). Jak si může pozorovatel všimnout ve vektoru 22b se nenachází sytě černá ani jasně bílá barva. Není tedy využit celý dynamický rozsah barev. Což je částečně způsobeno světelnými podmínkami. Vektory příznaků odebrané za různých světelných podmínek by se tedy od sebe výrazně lišily a mohly by být špatně rozpoznány. Vliv tohoto jevu se dá omezit normalizací příznakového vektoru (viz 2.1.3). Výsledkem je vektor využívající celého dynamického rozsahu intenzit (obr. 22c). Stejný postup je použit i na negativní vzorek (obr. 22f).

Tvorba trénovácí množiny

Po filtraci jsou příznakové vektory označeny jako pozitivní nebo negativní vzory. Takovýchto příkladů je vytvořeno dostatečně velké množství. Z nich se skládá trénovací množina pozitivních a negativních vzorů. Tyto množiny jsou následně spojeny do jedné trénovací množiny. Trénovací množina je promíchána, aby se při učení nevyskytovaly velké shluky stejného vzoru. Ty by mohly způsobit nevhodnou volbu dělící hyperplochy. V příkladu (obr. 23) jsou vizualizovány vybrané pozitivní (obr. 23a) i negativní (obr. 23b) příklady surových příznakových vektorů, před filtrováním.



(a) Pozitivní vzorky (b) Negativní vzorky

Obrázek 23: Příklady surových odebraných vzorků

Trénování SVM

Vytvořená množina příkladů je předána SVM. To pak provádí hledání hyperplochy dělící vzniklý mnohorozměrný prostor negativních a pozitivních vzorů postupem popsaným v kapitole (4.2.1). Po dokončení procesu učení je SVM serializováno (uloženo do XML souboru). Ze souboru je SVM načteno před procesem detekování závory v nových snímcích.

4.2.3 Detekce závory

Proces detekce závory vyžaduje natrénované SVM. Dále je třeba označit oblast, kde se ve snímku nalézá spuštěná závora. Tato informace je do programu předána ve formě parametrů příkazové řádky. Úsečka je definována dvěma koncovými body v dvou rozměrném prostoru obrazu. Dohromady jsou tedy předány čtyři hodnoty, definující souřadnice koncových bodů. Protože je kamera statická, stačí ruční označení polohy závory provést jednou. Po délce úsečky jsou odebrány vzorky barvy a zpracovány stejně jako v příkladu na (obr. 22). Samozřejmě musí být počet příznaků vektoru stejný pro jaký bylo natrénováno SVM. Získaný vektor příznaků je pak vyhodnocen pomocí SVM načteného z XML souboru. Pokud je vzorek klasifikován jako pozitivní vrací program detektoru hodnotu 1, jinak 0.

4.3 Porovnaní metod SVM a prahování

V této sekci srovnávám metody z hlediska implementačního, komfortu pro uživatele i přístupu k hledání závory. Uvadím taktéž očekávané slabé i silné stránky obou dektorů. Úspěšnost detekce pomocí vytvořených metod bude porovnána v následující kapitole experimentů.

Z hlediska složitosti implementace byla náročnější metoda využívající thresholding, která tvoří jeden program. V něm jsou spolu všechny části neoddělitelně propojeny. Samotný problém jak rozpoznat ve snímku přejezdovou závoru řeší autor. Oproti ní v metodě využívající strojové učení SVM se o rozpoznání přejezdové závory stará SVM, jehož implementaci poskytuje knihovna OpenCV. Celý proces je rozložen na několik samostatných modulů: tvorba trénovacích dat, vytrénovaní SVM, rozpoznání přejezdové závory.

Z pohledu uživatele je poněkud pracnější obsluhovat metodu využívající SVM. A to především v okamžiku tvorby příkladů pro učení. Tuto práci však může provést i autor

programu a zákazníka zásobit vytrénovaným SVM. Pak už je nutné jen označit pozici závory pro každou kameru, kterou chce uživatel k detekci využít, přičemž kamera musí být statická. Oproti tomu metoda využívající thresholdingu od koncového uživatele žádnou aktivitu uživatele pro samotný proces detekce nepotřebuje. Vyžaduje pouze aby kamera neležela na boku. Řečeno formálněji, aby pro člověka stojícího ve scéně byl směr vzhůru shodný se směrem k hornímu okraji obrazu. Pokud by tato podmínka splněna nebyla, lze jednoduše snímek před samotnou detekcí otočit do vyhovující polohy.

Při hledání závory v obraze prozkoumává metoda strojového učení pouze oblast označenou úsečkou. Může se tedy stát, že spuštěná přejezdová závora nemusí být detekována, dochází-li k silnému houpání závory například vlivem větru. Výhodou je, že i kdyby se ve scéně vyskytovaly další červenobíle pruhované objekty podobné závoře, nedojde k chybnému detekování spuštěné závory. Oproti tomu metoda využívající thresholding prochází celý snímek. Může se tedy stát, že mylně detekuje některý podobný objekt jako spuštěnou závoru. Na druhou stranu při mírném pohybu přejezdové závory je stále spuštěná závora detekována. A díky nastavení tolerance sklonu bývá závora detekována již při přiblížení k úhlu zaujímanému při spuštění.

5 Experimenty

Tato část se věnuje porovnání efektivnosti obou vytvořených detektorů přejezdových závor. Nejdříve jsou popsány vstupní snímky detekce a data získaná po provedení detekce. Tato data jsou následně analyzována vhodnými metodami. Výsledky experimentů jsou na závěr shrnuty, zhodnoceny a detektory porovnány.

5.1 Vstupní data

Pro testování obou detektorů jsem použil shodnou vstupní množinu snímku (příloha 2) ze staticky umístěné kamery sledující jednu stranu železničního přejezdu. Přejezd je umístěn ve volné krajině bez husté zástavby. Jedinou budovou v záběru je kontrolní stanice přejezdu. Sekvenci testovacích snímků jsem vybral tak, aby obsahovala zdviženou i spuštěnou závoru, pohyb za spuštěnou závorou, přejezd vlaku kolejištěm, proces spouštění závory i pohyb v kolejišti při zdvižených závorách. Pro rozpoznávání pomocí SVM jsem stanovil pozici závory ve scéně.

Obrázek 24: Příklad testovaného snímku

Snímek (obr. 24) je jedním příkladem z testovací množiny. Je mu přidělen unikátní identifikátor 000099. Přejezdové závory jsou spuštěny. Za závorami se nalézají další osoby, které by mohli ztížit proces detekce díky svým oranžovým výstražným vestám s bílými pruhy. Viditelnost na oblast přejezdu je dobrá. Jediné co by mohlo způsobovat problém je to, že slunce přímo neosvětluje stranu závory přilehlou ke kameře. Díky vzniklému stínu se bíl0 pruhy mohou jevit jako šedé a splývat s šedou sluncem zalitou vozovkou.

Všechny snímky z vytvořené testovací množiny byly zpracovány jak detektorem využívající SVM, tak detektorem využívajícím thresholding. Abych mohl zjistit úspěšnost detekce musel jsem také ručně projít množinu a zaznamenat výskyt závory ve snímcích. Takto jsem vytvořil referenční vzorek. Výsledky jsem zanesl do tabulek (příloha 2).

detekce závory			
id snímku	SVM	tresholding	ruční
	boolean	boolean	boolean
000082	FALSE	FALSE	FALSE
000083	FALSE	TRUE	FALSE
000084	FALSE	TRUE	TRUE

Tabulka 1: Příklad výsledků detekce pro použité metody

Příklad (tab. 1) představuje výřez obsahu tabulky výsledků detekce závor. První sloupec obsahuje jedinečné identifikátory snímků. Další sloupce obsahují výsledky detekce pomocí metod SVM, thresholdingu a ruční kontroly. Tyto sloupce nabývají dvě logické hodnoty: TRUE nebo FALSE. Hodnota TRUE znamená, že v daném obraze závora byla detekována. Hodnota FALSE je uvedena pokud se závoru nalézt nepodařilo. Při ruční kontrole jsem za nalezení závory považoval situaci, v níž byla alespoň jedna závora zcela viditelná a nebyla v poloze kolmé k vozovce. Tyto binární data dále analyzuji.

5.2 Analýza

Tato část textu se zabývá rozborem zjištěných výsledků klasifikací. Nejprve jsem zjišťoval úspěšnost obou detektorů pomocí absolutní a relativní četnosti úspěšných detekcí z celkového množství snímků.

detektor	počet snímků	absolutní četnost	relativní četnost
	-	-	-
svm	1000	991	0.99
thresholding	1000	997	1.00

Tabulka 2: Četnosti správných detekcí

Tabulka četnosti správných detekcí (tab. 2) uvádí ve sloupci absolutní četnost počet správně klasifikovaných snímků. Ve sloupci relativní četnost je počet správně klasifikovaných snímků vztažen k celkovému počtu snímků. K tomu bylo potřeba porovnat výsledky každého detektoru s referenčním vzorkem a sečíst případy ve kterých se vzorky shodovaly. Tím byla získána absolutní četnost správných detekcí.

Rozhodl jsem se při analýze využít také matthewův korelační koeficient (MCC), protože detektor přejezdových závor vrací binární výsledky a jsem schopen stanovit referenční vzorek.

Tento se často využívá ve strojovém učení jako míra kvality u binárních (dvou-třídových) klasifikací. Přičemž zjistím MCC jak pro detektor s SVM tak pro detektor s thresholdingem, abych mohl metody porovnat. Ke zjištění MCC (34) je nutné rozdělit klasifikace do čtyř skupin: TP pravdivě pozitivní, TN pravdivě negativní, FP falešně pozitivní a FN falešně negativní. Součet všech $TP+TN+FP+FN$ je roven počtu testovaných snímků. Zjistil jsem tedy četnosti případů v těchto skupinách samostatně pro každý vektor. Výraz (34) slouží k výpočtu MCC. Výsledky jsou vyneseny do tabulky 3.

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{34}$$

detektor	TP	TN	FP	FN	MCC
	-	-	-	-	
svm	112	879	1	8	0.96
thresholding	119	878	2	1	0.99

Tabulka 3: Zjištěné hodnoty MCC

Tabulka 3 uvádí zjištěné hodnoty MCC a absolutní četnosti nastalých detekčních stavů. Hodnoty MCC se pohybují v itervalu $< +1; -1 >$. Přičemž hodnota $+1$ znamená perfektní shodu klasifikace a referenční hodnoty, hodnota 0 náhodné výsledky a hodnota -1 totální rozpor mezi výsledkem klasifikace a referenční hodnotou.

5.3 Shrnutí

Zjištěné hodnoty MCC ukazují, že oba detektory pracují nad vybranými snímky dobře. Správně určují stav závory. Jedinými problémovými okamžiky jsou snímky, na nichž jsou závory spouštěny nebo zvedány a okamžiky přejezdu vlaku. V příkladu (obr. 25) jsou uvedeny snímky, u nichž byla nesprávně klasifikována přítomnost závory. Tyto případy budou podrobněji rozebrány dále.

(a) Pohyb závory (b) Citlivost kamery (c) Barva vlaku

Obrázek 25: Problémové situace

U detektoru využívajícího thresholding dochází k detekci závory i když je v obraze viditelná jen její část. Jedná se o dva případy FP. Zde záleží na dané aplikaci. U některých by bylo vhodné tyto případy eliminovat. U jiných ponechat a brát tyto případy jako správně detekovanou závoru. Při tvorbě referenčního vzorku jsem označil, že snímek obsahuje závoru jen tehdy pokud v něm byla viditelná většina závory a její konec.

Druhým a posledním problémovým případem tresholdigu byl okamžik po přejezdu vlaku. Kvůli světelnému přechodu byla kamera citlivější na světlo a scéna je přesvícena (obr. 25b). Díky tomu se silnice jeví jako bílá a splývá s bílými pruhy závory. Což znemožnilo nalezení objektu závory jako celku. Došlo tedy k falešně negativní klasifikaci FN.

Detektor využívající SVM má problémy detekovat závoru mimo koncovou spuštěnou pozici (obr. 25a). Což je způsobeno tím, že je prozkoumávána jen oblast ležící na vybrané úsečce. Tento problém se projevil na osmi snímcích jako FN. Jestli je třeba tyto případy eliminovat nebo naopak přijmout za správné záleží na konkrétní aplikaci.

Zajímavý problém se projevil na snímku, v němž přes přejezd přejížděla vlaková souprava. Její světlý nátěr v oblasti dveří spolu s tmavým těsněním okénka vytvořil po převedení do odstínů šedi vzor podobný pruhování závory (obr. 25c). Během přejezdu vlaku byl v ten "pravý"okamžik odebrán vzorek pro klasifikaci právě z tohoto místa. Díky tomu SVM vyhodnotilo snímek jako obsahující závoru. V referenčních vzorku jsem však při přejezdu vlaku označil, že ve snímcích závora není. A to z prostého důvodu, že není viditelná.

6 Závěr

Při tvorbě této diplomové práce jsem získal nové poznatky a užitečné zkušenosti z oblasti počítačového vidění. Nejprve jsem rozebral chyby vyskytující se ve vstupním obraze a způsoby jejich minimalizace. Nastudoval jsem také problematiku detekce objektů a související oblast strojového učení využívanou pro klasifikaci objektů. Osvojené dovednosti jsem následně využil pro navrhnutí autonomní kontroly přejezdových závor.

Při navrhování autonomní kontroly přejezdových závor jsem se rozhodl vytvořit dva detektory pomocí odlišných přístupů k problému rozpoznávání přejezdové závory. První přístup prohledává celý snímek. Druhý testuje pouze vybranou část snímku za pomoci strojového učení. V případě druhého přístupu je nutné, aby uživatel vybral oblast v níž se závora nachází. A to pro každou používanou kameru. Kamery jsou však statické. Proto je tento úkon nutné provést jen jednou. Porovnání účinnosti obou detektorů jsem provedl v sekci experimentů. Ukázalo se, že oba detektory pracují dobře. Na testovacích datech přesáhly hodnoty matthewůova korelačního koeficientu každého detektoru hodnotu 0.95. Rozdílné metody detekce se však projevily právě na tom, které snímky byli špatně klasifikovány.

Postup při němž je prohledáván celý snímek má tu nevýhodu, že může být ovlivněn nedokonalostmi techniky, jako sklony k přesvětlení snímku při rychlých změnách stínu a prudkého světla dopadajících na snímač obrazu. Taktéž může být zmaten objekty podobnými závoře a chybně tak detekovat její přítomnost. Oproti tomu druhý postup prohledávající jen vyznačenou oblast tímto neduhem netrpí. Za to, ale nedokáže detekovat přejezdovou závoru při pohybu mezi zdviženou a spuštěnou polohou. Ovšem i tento detektor může být oklamán pokud se v prohledávané oblasti vyskytuje pruhovaný objekt. Počet chybně klasifikovaných snímků však nepřesáhl 1% z celkového množství 1000 testovaných snímků. Proto lze tyto případy považovat za ojedinělé.

Pro vizuální potvrzení spuštěné závory jsou problémovými okamžiky přejezdy vlaku. Během nich není závora viditelná. Tyto snímky oba detektory vyhodnotí jako bez závory. Což je správný výsledek. I když člověku je jasné, že závora je spuštěna i přesto, že není viditelná. Závory jsou však spouštěny dříve než vlak dorazí k přejezdu. Proto může být jejich stav potvrzen před přejezdem vlaku. Primární účel kontroly stavu závory tak zůstává splněn.

Celkově hodnotím oba detektory velmi dobře a při praktickém nasazení může být vybrán ten, jehož chování v mezních situacích lépe vyhovuje dané úloze. Případně mohou být využity oba. Přičemž konečný výsledek klasifikace může být získán pomocí operací logického součtu či součinu, dle dané úlohy. Pevně věřím, že vytvořené detektory stavu přejezdových závor zvýší bezpečnost železničních přejezdů a zachrání tak lidské životy.

Reference

[1] Eduard Sojka, *"Digitální zpracování a analýza obrazu"*, VŠB-TU Ostrava, 2000

[2] L. G. Shapiro and G. C. Stockman *"Computer Vision"*, Prentice Hall, 2001

[3] M. Gluč, *"Edge Road Detection"*, VŠB-TU Ostrava, 2012

[4] G. Bradski and A. Kaehler, *"Learning OpenCV"*, Inc. O'Reilly Media, 2008

[5] H. J. Haverkort, *"Results on geometric networks and data structures"*, Universiteit Utrecht, 2004

[6] Rafael C. González, Richard Eugene Woods . *"Digital Image Processing*, Prentice Hall, 2007

[7] Edward R. Dougherty, *"Mathematical Morphology in Image Processing"*, CRC Press, 1992

[8] C. Cortes and V. Vapnik, *"Support-vector networks"*, Machine Learning, Volume 20, Issue 3, str 273-297, 1995

[9] R. A. Jarvis, *"On the identification of the convex hull of a finite set of points in the plane"*, Information Processing Letters, Volume 2, str 18–21, 1973

[10] Volkswagen, Laboratoř elektronického výzkumu, *"VOLKSWAGEN Group of America, Inc. Electronics Research Laboratory"* [online]
`http://www.vwerl.com`

[11] K. Bonsor, *"How Facial Recognition Systems Work"* [online]
`http://computer.howstuffworks.com/facial-recognition.htm`

[12] Edward H. Adelson, *"Osobní stránky Edward H. Adelson"* [online]
`http://persci.mit.edu/people/adelson`

[13] Knihovna OpenCV, *"OpenCV* [online]
`http://opencv.org`

Seznam příloh

1 CD s kompletním obsahem práce a detektorem

2 datová příloha v archivu ZIP